LEVERAGING IMAGE MANIFOLDS FOR VISUAL LEARNING

By

AMR M. BAKRY

A dissertation submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Doctor of Philosophy

Graduate Program in Computer Science

Written under the direction of

Ahmed Elgammal

And approved by

_____

_____

_____

_____

New Brunswick, New Jersey

October, 2016

ABSTRACT OF THE DISSERTATION

Leveraging Image Manifolds for Visual Learning

By Amr M. Bakry

Dissertation Director: Ahmed Elgammal

The field of computer vision has recently witnessed remarkable progress, due mainly to visual data availability and machine learning advances. Modeling the visual data is challenging due to several factors, such as loss of information while projecting 3D world to 2D plain, high dimensionality of the visual data, and existence of nuisance parameters such as occlusion, clutter, illumination and noise. In this dissertation, we focus on modeling the inter and intra image manifold variability.

The dissertation shows that modeling the image manifold helps to achieve recognition invariance and perform robust regression within the manifold. It leverages the power of Homeomorphic Manifold Analysis (HMA) framework to utilize the known topological information about data manifolds. HMA builds mappings from a conceptual space to the feature space. These mappings are based on topological homeomorphism between points in the two spaces.

The dissertation extends this framework, applied to several applications such as human motion analysis and object recognition in conjunction with pose estimation. We propose Manifold-KPLS (MKPLS), a discriminative nonlinear model for recognition of motion sequences, applied to visual speech recognition. To tackle recognition and pose estimation from single test image, we propose a bi-nonlinear generative framework. The dissertation uses iterative inference techniques to find the optimal category and viewpoint that match a given test image. To speed up the inference, the dissertation proposes a feed-forward model, which is more efficient and more accurate for solving the same problem. On the other hand, the dissertation leverages the manifold analysis to propose quantitative measurements for building a CNN variant for simultaneously solving object recognition and pose estimation.

Table of Contents

List of Tables

List of Figures

Chapter 1
Introduction

1.1 Motivation

Influenced by the recent advances in machine learning, computer vision is witnessing remarkable revolution. The gap between theoretical research and real-life applications is diminishing. However, working with data in the image feature space is still challenging due to the large variability in images which affects the dimensionality and distribution of points in the image feature space. The appearance of objects in images is sensitive to many factors of variation such as category, viewpoint, scale, deformation, illumination, noise, occlusion and background clutter. Depending on the application, few of these factors are important and most of them are nuisance. To claim that we have robust machine vision framework, it has to be invariant to the nuisance factors and correctly encodes the important factors.

Even though, the field of computer vision has recently been enriched with huge datasets [Deng et al., 2009; Lin et al., 2014], it is not guaranteed to have sufficient sampling for the important factors of variations. Therefore, robust modeling for visual data is essential for generalizing the available samples in a way that can help for invariant and efficient recognition.

Figure 1.1 illustrates the meaning and importance of data modeling and manifold analysis. Figure 1.1.a shows a perfect 2-D swiss-roll manifold in

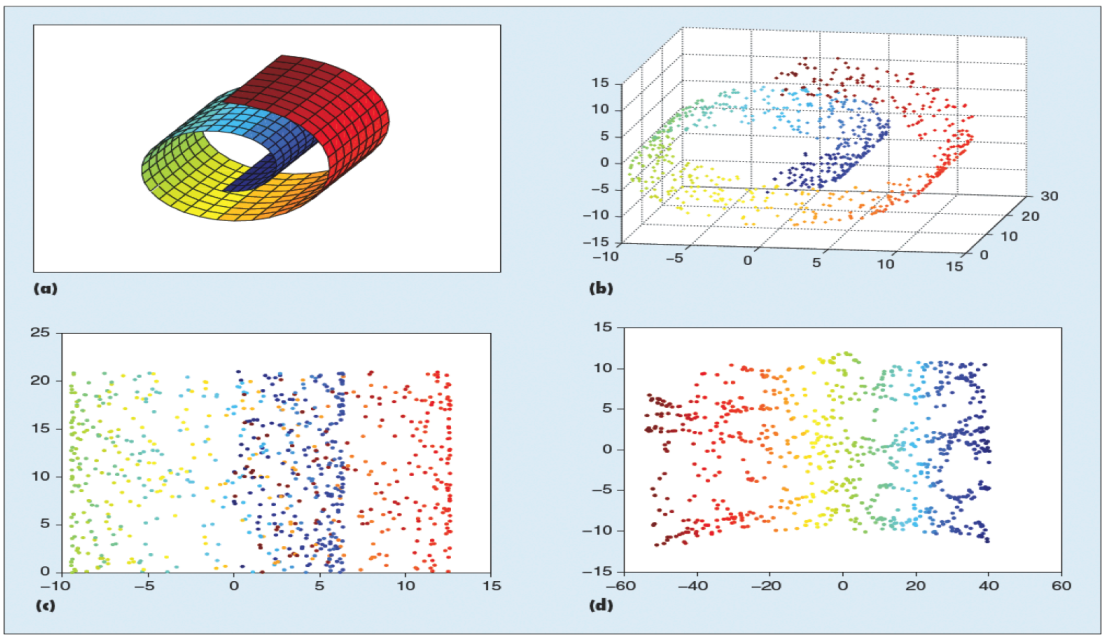


Figure 1.1: Swiss-roll manifold as presented by [Roweis and Saul, 2000]

higher-dimensional space. Figure 1.1.b shows sampled points on this manifold represented in a 3-D Euclidean space. As we can see, the points are sparsely sampled with remarkable sampling noise. For instance, if colors encodes different values for a scalar function, the question is, can we model the data in order to perform robust regression. Robust means that the modeling reflects the actual topology on the manifold and is invariant to the sampling noise. Figure 1.1.c shows that using linear dimensionality embedding of the points in PCA space is not robust, because the neighborhood relationship between the points is not preserved. But when using Locally Linear Embedding (LLE) [Roweis and Saul, 2000] which utilizes the prior neighborhood information to perform the embedding, we get a much better and relevant low-dimensional representation of the points, as in Figure 1.1.d. This shows that considering the neighborhood knowledge is important to model data variability in high-dimensional space.

Data modeling is a fundamental problem in machine learning. With the recent advances in computational platform and availability of large datasets such as ImageNet [Deng et al., 2009] and Microsoft Coco [Lin et al., 2014], Neural Networks (NN) are progressing significantly and their deep versions are widely used for learning representations of the variations in the feature space. Convolutional Neural Networks (CNN) [LeCun et al., 1998] were proposed for handwritten character recognition in images. CNN is superior for achieving invariant recognition, however its ability for generalizing the available data samples is questionable. On the other side, Gaussian Processes (GP) [Rasmussen and Williams, 2006] is a different modeling technique. The main advantage of GP modeling is its generalization ability and accurate modeling around the available data samples. However its scalability is limited, since it uses the complete training dataset for inference.

Riemannian geometry has been widely explored to model the visual data manifolds. Riemannian manifold gives a concrete way to model the geodesic path and distance between images as matrices, so that we do not have to vectorize the images to work in vector space. For example, Freifeld and Black [2012] used Riemannian manifold to model the variation in 3D human shape space. In this work, shape is represented by a Lie group. Azary and Savakis [2013] used Riemannian manifold of subspaces (Grassmannian manifold) to model the variations within ensembles of depth image. They used this for 3D action recognition. Elgammal and Lee [2004a] proposed Homomorphic manifold analysis (HMA) to model the variations in the human motion space.

Figure 1.2 shows sampled points from view manifolds of car object and iron object. View manifold is the trajectory of changing the viewpoint of the same object in the image feature space. Figure 1.2 also shows sampled points from

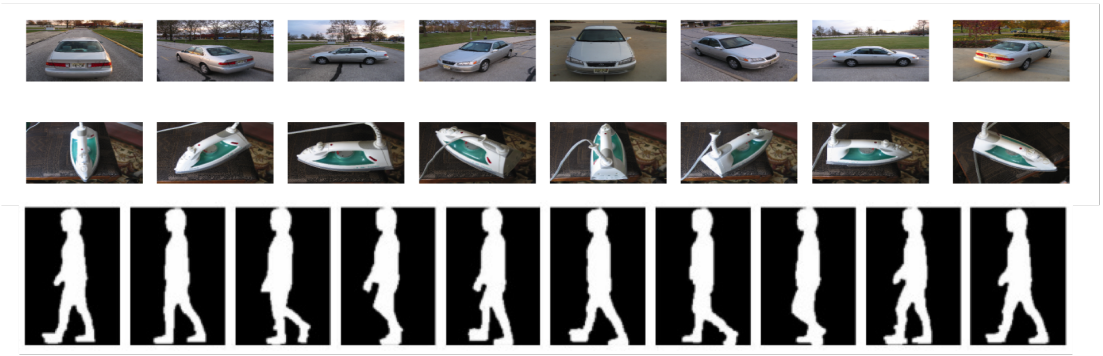


Figure 1.2: Sampled points on object-view manifold (First two rows), and motion manifold (third row).

a manifold of human motion. In these examples, we know the ideal neighborhood relationship between these images. In other words, we know the exact ordering of the these images on the latent manifolds.

In the work by Donoho and Grimes [2005], the authors showed that the image manifold is not locally smooth nor differentiable. Even for pure images with white foreground and black background the smoothness is not guaranteed due to existence of edges. In this work they proposed extension for LLE named by Hessian Locally Embedding (HLE) to tackle this problem in image feature manifold. It is worth mentioning that local smoothness and differentiability are basic assumptions behind the theory of Riemannian manifolds [Tenenbaum et al., 2000; Donoho and Grimes, 2005].

Figure 1.3 shows the image feature space from the point of view of instance manifolds. It shows a set of single dimensional manifolds connecting images in the feature space. Everyone of these structures represents a single instance manifold. It could be an instance of an object-view manifold, an instance of a motion manifold, or any other appropriate manifold. The question is how to utilize this knowledge to build a framework that is able to discriminate between different manifolds or to recognize the information in a single image.

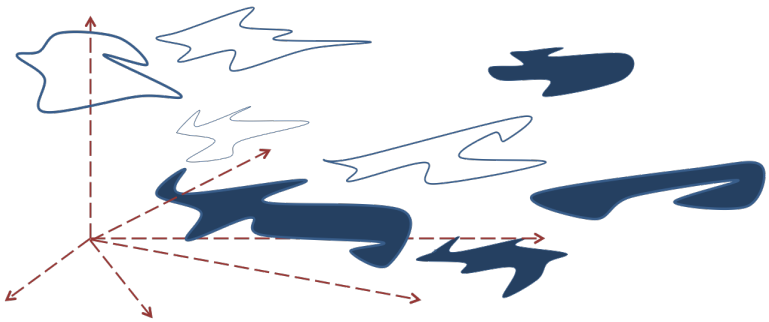


Figure 1.3: Instance manifold visualization in feature space.

In this dissertation, we extend the Homomorphic Manifold Analysis (HMA) framework, proposed in [Elgammal and Lee, 2013] to encode the topological/neighborhood information between images in sets of image-ensembles to build a robust visual model. HMA is a pipeline of two phases. The first phase is parameterizing the instance manifolds, and the second one is dimensionality reduction by style factorization. More details about HMA is in Section 2.1.

1.2 Contributions overview

The contribution of this dissertation is: proposing several extensions to the HMA framework and proposing a probabilistic generalization to HMA. The dissertation also uses manifold analysis to achieve deeper understanding of CNN. In this section, we highlight these contributions.

1.2.1 Supervised nonlinear model for manifold discrimination

The dissertation proposes Manifold Kernel Partial Least Squares (MKPLS) framework which is intended to model the variations between complete instance

manifolds, as illustrated in Figure 3.1. MKPLS is an extension to HMA. It parameterizes the manifolds using Gaussian Radial Basis Functions (Gaussian RBF), and it customizes Kernel Partial Least Squares (KPLS) [Rosipal and Trejo, 2002] for embedding the manifold parameterizations into low-dimensional latent space. KPLS is a nonlinear supervised dimensionality reduction technique. It compromises between minimizing the reconstruction error and maximizing cross correlation between latent points and its corresponding labels. More details about KPLS can be found in Chapter 2.2. MKPLS is intended to discriminate between different manifolds. The novelty here is applying KPLS on instance manifolds, based on its parameterization, to reach a latent space where each manifold is represented by a point. This is in contrast to traditional usage of KPLS on data points, where each image is a point.

We apply MKPLS on visual speech recognition to recognize the spoken word in utterance footage and to identify speaker. Visual speech recognition is a challenging problem, due to the confusion between visual speech features. The speaker identification problem is usually coupled with speech recognition. Speaker identification is important to several applications, such as automatic access control, biometrics, authentication, and personal privacy issues. We initially parameterize the instance manifold of each uttered video using a nonlinear mapping from a unified manifold representation. We then factorize the parameter space using KPLS to achieve a low-dimension manifold latent space. We use two-way projections to achieve two manifold latent spaces, one for the speech content and one for the speaker. We apply our approach on two public databases, and we show the results for three different settings of lipreading: speaker independent, speaker dependent, and speaker semi-dependent. Our approach outperforms for the speaker semi-dependent setting by at least 15%

of the baseline, and competes in the other two settings. The details of this framework and its experiments and the results are described in Chapter 3.

1.2.2 Bi-nonlinear generative model for image based inference

Object recognition and pose estimation are two fundamental problems in the field of computer vision. Recognizing objects and identifying their poses (viewpoints) are critical components of vision and robotic systems. The shape and appearance of an object in a given image is a function of its category, style within category, viewpoint, and several other factors. The visual manifold (in any chosen feature representation space) given all these variability collectively is very hard and even impossible to model.

In this part of the dissertation, we consider the problem of modeling the combined object-viewpoint manifold. Multiple viewpoints of an object lie on an intrinsic low-dimensional manifold in the input space (*i.e.* feature space). Different objects captured from the same set of viewpoints have manifolds with a common topology, as exemplified by Figure 1.2. Based on this fact, we propose using a unified viewpoint space as a common representation for all object-view manifolds. Therefore, we can parameterize every instance manifold in terms of points in this unified latent space. By embedding the instances' parameterization, we get low-dimensional *style* latent space as a view-invariant category representation. We end up with a bi-nonlinear generative model that maps points from the viewpoint latent space and the style latent space to points in the image feature space.

Based on this model, the dissertation utilizes the MKPLS framework, Section 1.2.1, for image based analysis, as follows. For doing this, we augment MKPLS with a reverse map from the style latent space to the feature space.

This augmentation is possible because KPLS is maintaining low reconstruction error as well as encoding supervision.

For inference, we explore the style space and the viewpoint space looking for predicted image which is closest to the test image. Since the two latent spaces are continuous, we use iterative techniques for inferences such as MCMC sampling and Gradient descent. The advantage of continuous inference is the scalability, and iterative inference helps to correct initial or intermediate wrong latent positioning. We empirically validate our model by testing on multiple challenging datasets. We compare our results with the state-of-the-art and present our increased category recognition and pose estimation accuracy.

1.2.3 Feedforward model for efficient inference

Despite the advantages of using continuous iterative methods for inference, in Section 1.2.2, it is computationally expensive. Therefore, the dissertation proposes an efficient computational framework that can untangle such a complex object-view manifold, and achieve a model that separates a view-invariant category representation, from category-invariant pose representation. The proposed cascaded framework is illustrated in Figure 1.4 in a deep network formate. Similar to the model described in Section 1.2.2, the proposed model is based on the same two latent spaces (viewpoint space and style space) . However, unlike the bi-nonlinear model, the style latent space is learned with an unsupervised linear dimensionality reduction technique (PCA) [Zhang et al., 2013]. It is worth mentioning that the proposed framework is orthogonal to the selected feature extractor. In the experiments, we use HoG features [Dalal and Triggs, 2005]. However, this can be extended to any other appropriate feature

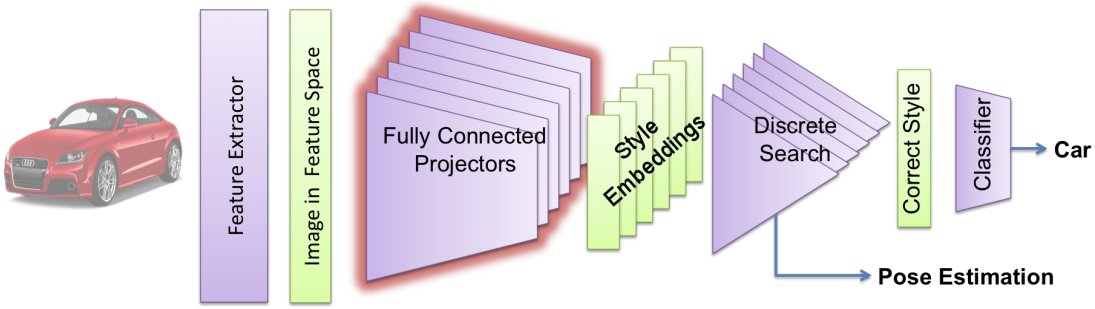


Figure 1.4: This cartoon illustrate the procedures in the proposed Feedforward framework for joint object-recognition and pose-estimation. After extracting, the framework projects the input image to the latent style space. Then, using a novel objective function the framework can decide the correct style embedding. This selection leads directly to pose-estimation, and classifying the style embedding leads to object-recognition. The details are in Chapter 5.

extractor such as convolutional network features.

To that end, we replaced the continuous viewpoint latent space by discrete grid. This step is feasible since the geometry of the viewpoint space is known. For instance, the single degree object-view manifold is homomorphic to a circle. Two dimensional view space is homomorphic to sphere, and higher dimensional view space is homomorphic to hyper sphere. At every point of the viewpoint grid, we learn a view projector, see Figure 1.4. Every projector maps the input image into a style embedding. Since the projectors are function of the viewpoint, not all of them are consistent with the same image. This means that only one of the resulting style embeddings is consistent with the input image. In order to identify the correct embedding, the dissertation derives a objective function. The result of this function is index of the correct style embedding. By selecting the correct embedding, we implicitly have estimation for the viewpoint. Thence, we apply the pre-trained category classifier to identify the object in the test image. It turns out that the proposed framework is not only more efficient but also more accurate. We outperform the state-of-the-art

result in three widely used multi-view dataset, for both category recognition, and pose estimation. To leverage the accurate and efficient performance of this framework, The dissertation briefly discusses how to perform the object detection, in conjunction with over-the-shelf detection techniques. Chapter 5 provides a complete details of the proposed framework and the conducted experiments.

1.2.4 Probabilistic bi-nonlinear generative framework for unified content and style discrimination

Modeling intrinsic variability in visual data in the presence of multiple styles is essential and challenging problem. For doing this, the dissertation proposes a probabilistic bi-nonlinear generative model based on Gaussian Processes (GP) [Rasmussen and Williams, 2006]. The dissertation leverages the power of GP to find a unified content latent space and discriminate the styles, for several computer vision applications.

The main contributions in this work: we learn the unified latent space for the instance manifolds instead of setting it manually. We also derived a closed form for the joint likelihood of the feature space as a function of multiple orthogonal low-dimensional latent spaces. We use this to propose a novel probabilistic nonlinear framework for embedding visual data into *content* and *style* latent spaces. To deal with the complexity and high-dimensionality of the visual data, the dissertation proposes a fast iterative algorithm that builds on top of the Gaussian Process Latent Variable model (GPLVM) [Lawrence, 2005].

In the specific case of human motion analysis, data is composed of several instance manifolds of motion sequences. The proposed model helps to

find the common topological manifold of these motion sequences (content embedding). Moreover, our proposed framework embeds every sequence into a style latent space, where it is represented by a single point (style embedding). The topological content embedding is useful for the tracking the moving subject, while the style embedding helps to discriminate between different motion style such as walking, jogging and running. Both latent embeddings can be used to generating motion sequences with different pace or style.

Gaussian Process Dynamical Model framework (GPDM) [Wang et al., 2005] is proposed to find low-dimensional latent trajectory for a single motion sequence. However, for multiple sequences, GPDM produces significantly different embeddings even for sequences of similar styles such as walking. This limits the robustness for any further inference based on the the generated embedding. Therefore, Balanced-GPDM [Urtasun et al., 2006] adds more priors regularize embedding the manifolds. This helps the authors to use the mean embedding trajectory to track the human motion [Urtasun et al., 2006], and infer the body pose. This illustrates the contribution of our framework, which produces a unified content space (pose trajectory) that helps to track the motion, and utilizes the dynamical variations to find more robust style representations as well.

For inference, the probabilistic modeling gives a concrete way to find the best solution for the objective function. To illustrate the effectiveness, we apply the proposed framework on a toy dataset, as well as several publicly available computer vision datasets. The experimental results and details of this work is presented in Chapter 6.

1.2.5 Understanding view-invariance in CNN

For enriching the field of deep learning and for supporting the efforts to understand what is happening inside deep networks, the dissertation designates Chapter 7 to study the feasibility of multi-task learning of the Convolutional Neural Network (CNN) for solve the object recognition and pose estimation jointly. These two problems are coined together because of the aforementioned variability in the image feature space and because they have special interest in many applications. In our study, we analyze the transformations happening to object view manifold through out the layers of AlexNet [Krizhevsky et al., 2012] [1]. More specifically, we are looking for the layers that have the object's view manifold is well preserved (for pose estimation) and well separated for different classes (for categorization). For measuring these two aspects (separability and preservability), the dissertation proposes a set of quantitative measurements. The measurements are based on well founded machine learning concepts and reasonable intuitions. The dissertation proposes a synthetic dataset to verify the efficiency of these measurements and to make sure that it is qualified to be used as a fair benchmark to monitor view manifolds in different feature spaces.

The dissertation uses these measurements as a tool to monitor the two previously discussed aspects in CNN. By applying these measurements to all layers in AlexNet, we extract sets of conclusions. The dissertation provides a set of conclusions that are intended to light up the black box of the convolutional neural networks.

Among many of the observations and conclusions, we observed that as we

[1] AlexNet has five convolutional layers and three fully connected layers named Fc6, FC7 and FC8

go deeper in convolutional layers, the separability and preservability of the object view manifolds are getting better. Then as we go further deeper in the fully connected layers we gain more separability at the expense of the preservability. More specifically, layer *Pool 5* in AlexNet has the best balance between separability and preservability of the object view manifolds. This tells us that if we want to solve the categorization and pose estimation jointly, we want to extract features from the last convolutional layer. For details, refer to Chapter 7.

1.3 Associated Publications

Part of the work in this dissertation has been published in several conferences. The work in Chapter 3 has been published in [Bakry and Elgammal, 2013; Bakry and Elgammal, 2016]. While, the text in Chapter 5 borrows from [Bakry and Elgammal, 2014]. The work in Chapter 4 is published in[Bakry et al., 2016]. The work in Chapter 7 is published in ICLR 2016 [Bakry et al., 2015]. The work in Bakry et al. [2016, 2015] are co-authored by Tarek Elgaaly and Mohamed Elhosieny. The work in Chapter 3 was partly supported by the National Science Foundation award number 0923658 and by the Office of Navel Research grant N00014-12-1-0755.

Chapter 2

Background

2.1 Homomorphic Manifold Analysis

Homomorphic Manifold Analysis (*HMA*) [Elgammal and Lee, 2013] is proposed to encode the topological/neighborhood relationship in image ensembles. This framework utilizes the homeomorphism between points in a conceptual space and the manifold in the feature space. It is proposed as a cascaded pipeline of two phases: the manifold parameterization and style factorization, Figure 2.1.

HMA has already been applied in different computer vision applications: the basic idea of the HMA framework is introduced in the context of human motion analysis [Elgammal and Lee, 2004a], as a way to separate style and content on manifolds. Then, it is applied to different problems in the context

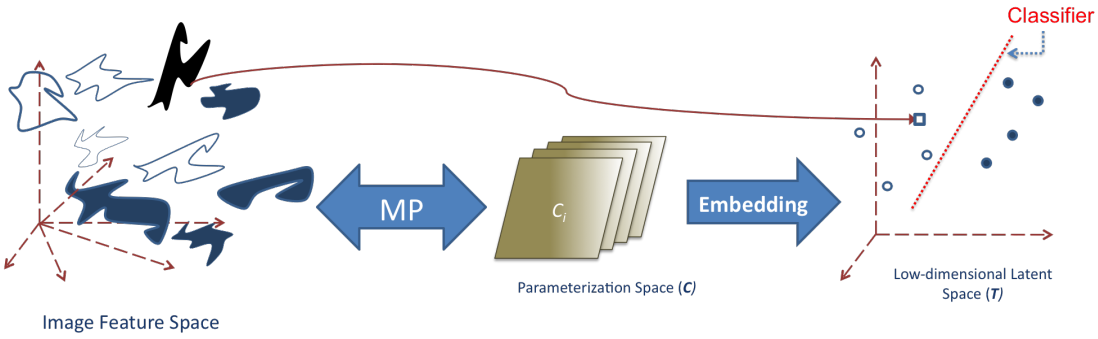
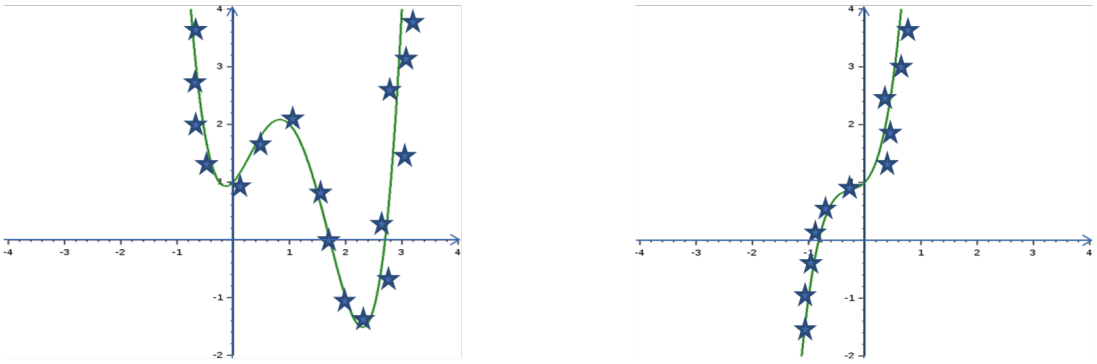


Figure 2.1: Instance manifold visualization in feature space.

of human motion analysis in different settings, including locomotion [Elgammal and Lee, 2004a,b, 2009] and facial expressions [Lee and Elgammal, 2005a]. HMA framework is also used for modeling two-dimensional manifolds within the context of modeling the joint configuration-view manifold [Elgammal and Lee, 2009], and in modeling complex human motions (such as ballet motion), by jointly modeled the kinematic and visual manifolds [Lee and Elgammal, 2010]. Recently HMA framework is used to model object-view manifold in the context of multi-view object recognition and pose estimation [Zhang et al., 2013].

2.1.1 Instance manifold

In all these applications, the dataset can be described as disjoint sets of *instances*. Points in each instance lie on a low-dimensional manifold that lives in arbitrary dimensional feature space. In other word, the data lies on multiple instances of manifolds sharing the same topology but differ in the geometry. For instance, Figure 1.2 shows three different instances for car, iron and motion manifold. In any of these applications, all instance manifolds are topologically equivalent, and homeomorphic to each other. A function $f : X \rightarrow Y$ between two topological spaces is called a homeomorphism if it is a bijection, continuous, and its inverse is continuous. This is correct except in the degeneracy case, where the instance manifolds are self-collapsing or self-intersecting. In high-dimensional feature space, this case is high unlikely.



$y = x^4 - 4x^3 + 3x^2 + x + 1$ $\qquad$ $y = 0x^4 + 4x^3 + 3x^2 + x + 1$

Figure 2.2: Manifold parameterization using polynomial basis

2.1.2 Individual Manifold Parameterization

The first step in HMA is to parameterize the image instance manifolds to obtain a descriptor for each of them. The manifold parameterization is first introduced in [Elgammal and Lee, 2004a; Lee and Elgammal, 2005b]. Figure 2.2 illustrates the meaning of manifold parameterization in general. In this figure, two instance manifolds are given and the goal is to find a concise way to represent the manifolds in order to encode the underlying topology. This representation might be used to discriminate between different instances. The two instances are single dimensional instance manifolds in 2D Euclidean space. Figure 2.2 shows that using a set of polynomial basis $\{x^0, x^2 \dots x^4\}$, we are able to find concise (5 dimensional) representation and precise so that it encodes all the information needed to reconstruct the manifolds accurately.

Figure 2.3 illustrates how HMA utilizes homomorphism between all instance manifold. It uses a low-dimensional conceptual *unified manifold*, which encodes the common topology of all instance manifolds. HMA learns regularized mapping function from this unified manifold to every individual instance manifold in the image feature space. The parameterization of these mappings

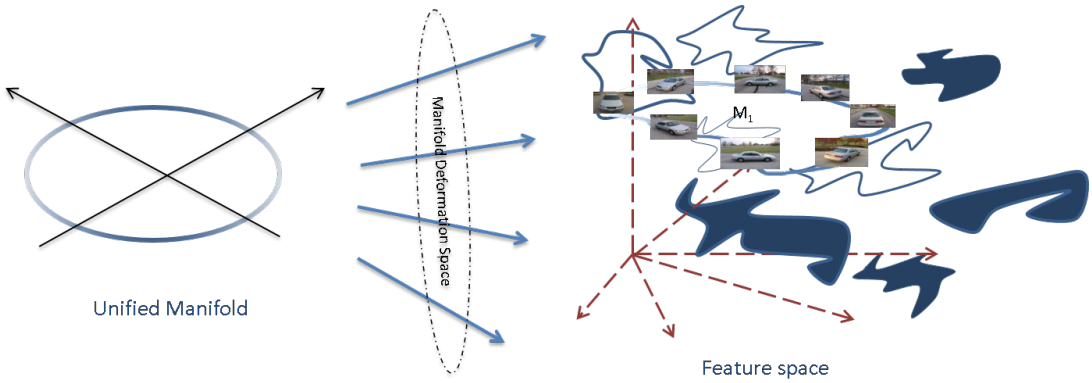


Figure 2.3: Homomorphic mapping from unified manifold and instance manifold.

are meant to encode the geometric deformations of the instance manifolds. Therefore, the space of these parameterization is a key component of HMA, since it simplifies the manifold representation and it helps to factor out some nuisance parameters.

It worth mentioning that the unified manifold representation is supposed to be topological equivalent to each instance manifold. This can not simply be obtained by traditional Dimensionality Reduction (DR) on the whole input data. This is because the goal of DR approaches is to find an embedding that preserves the local (or global) geometry of the data. In contrast, the unified manifold representation is a collapsing of all instance manifolds to one average manifold. There are various ways that can be used to achieve this. In [Elgammal and Lee, 2004a] individual manifolds are embedded and warped to compute an average embedding. Alternatively, if the topology of the manifold is known, a conceptual representation can be imposed; for example a unit circle can be used as topologically equivalent representation of all closed one-dimensional manifolds [Lee and Elgammal, 2005b]. Another alternative is to use manifold alignment (*e.g.* [Ham et al., 2005]) to learn a unified embedding.

Mathematically, Let $\{\mathbf{x}_i^k \in \mathbb{R}^D, i = 1, \cdots, n_k\}$ be the input images for

instance manifold $\mathcal{M}_k$, represented in a D-dimensional feature space. Let $\{\mathbf{z}_i^k \in \mathbb{R}^e, i = 1, \cdots, n_k\}$ be the corresponding embedded representation in an e-dimensional Euclidean space, which lie on the unified manifold $\mathcal{U}$. Notice that the number of points in each instance manifold does not need to be equal.

We learn mapping functions $\gamma^k(\cdot) : \mathbb{R}^e \to \mathbb{R}^D$, which maps from $\mathcal{U}$ to each instance manifold $\mathcal{M}_k$. To learn such mappings, we learn individual functions $\gamma_l^k : \mathbb{R}^e \to \mathbb{R}$ for the l-th dimension in the feature space. Each of these functions minimizes a regularized loss functional in the form

$$\sum_i^{n^k} \left\| \mathbf{x}_{il}^k - \gamma_l^k(\mathbf{z}_i^k) \right\|^2 + \lambda \, \Omega[\gamma_l^k], \tag{2.1}$$

where $\|\cdot\|$ is the Euclidean norm, Ω is a regularization function that enforces the smoothness in the learned function, From the representer theorem [Kimeldorf and Wahba, 1970; Poggio and Girosi, 1990] we know that such mapping functions admit a representation in the form of a linear combination of kernel basis functions in the embedding space $\mathbb{R}^e$. To achieve a common parameterization space of all the manifold, we use the same set of basis functions $K(\cdot, w_i), i = 1 \cdots n$, where $w_i \in \mathbb{R}^e$. The whole mapping can be written in the matrix form as

$$\gamma^k(\mathbf{z}) = \mathbf{C}_k \psi(\mathbf{z})$$

where $\mathbf{C}_k$ is a $D \times n$ matrix, and the vector $\psi(\mathbf{z}) = [K(\mathbf{z}, \mathbf{w}_1), \cdots, K(\mathbf{z}, \mathbf{w}_n)]$ represents a nonlinear kernel map from the embedded representation to a kernel induced space.

2.1.3 Low-dimensional Embedding

The obtained parameterizations live in a high-dimensional space, which makes it hard to learn classification functions that can generalize well. Therefore, low-dimensionality embedding is a key step in HMA. This step helps to learn concise representation of the manifolds, that encodes the differences in geometric deformations. This embedding is called *style space*. The final framework depends on the technique used for embedding the manifold parameterization. For instance, if embedding function is bijective this leads to generative model. In all the aforementioned HMA previous work , linear PCA subspace analysis is used to obtain a latent representation of the manifold parameterization space.

2.2 Kernel Partial Least Squares

Projection of data to a low-dimensional latent space is widely used in pattern classificaiton problems. The most common techniques for projection to a latent spaces are PCA and LDA [Duda et al., 2001]. Another technique that is widely used in chemometric pattern recognition is Partial Least Squares (PLS) [Wold, 1975; Rosipal and Trejo, 2002; Bennett and Embrechts, 2003]. Projection using PCA tends to keep most of the variance of the input space. In contrast, LDA tends to increase the clustering ability between different classes by maximizing the interclass and minimizing the intraclass distances [Duda et al., 2001]. PLS compromises by creating orthogonal components (in the latent space) using the existing correlations between explanatory variables (in the input space) and corresponding labeling, while keeping most of the variance of the points in the input space. A good interpretation for PLS and its relationship with

iterative PCA can be found in [Lewi, 1995; Bennett and Embrechts, 2003]. Additionally, PLS has been proven to be useful in situations where the number of the explanatory variables (dimensionality of the input space) exceeds significantly the number of observations and/or a high level of multicollinearity[1] among those variables.

For understanding the PLS, synopsis for PLS analysis [Bennett and Embrechts, 2003] is presented here. PLS is a least squares regression-based technique. Like PCA regression (PCR), PLS finds a regressor $\mathbf{w}$, so that, $y_i \approxeq \mathbf{x_i}^\top \mathbf{w}, \forall i$, where $\mathbf{x_i}$ is the observation and y_i is its response (output). If we put that in a matrix form, the objective is to minimize the least squares error $\|\mathbf{XW} - \mathbf{y}\|^2$. Bennett [Bennett and Embrechts, 2003] showed that

$$\|\mathbf{XW} - \mathbf{y}\|^2 \leq \|\mathbf{X} - \mathbf{yW}\|^2 .$$

Therefore, if we minimize $\leq \|\mathbf{X} - \mathbf{yW}\|^2$, we satisfy the objective. Then, he shows that

$$\min_{\mathbf{W}} \|\mathbf{X} - \mathbf{yW}\|^2 \propto \max_{\mathbf{W}} cov(\mathbf{XW}, \mathbf{y}), s.t. \mathbf{W}^\top \mathbf{W} = \mathbf{I}, \tag{2.2}$$

where *cov* stands for covariance. The solution of the Eq 2.2 has been shown to be

$$\mathbf{W} = \frac{\mathbf{X}^\top \mathbf{y}}{\mathbf{y}^\top \mathbf{X}\mathbf{X}^\top \mathbf{y}}, \tag{2.3}$$

which provides a closed form for $\mathbf{W}$.

However, for the high-dimensional observation space, Eq 2.3 is not robust and computationally inefficient. On the other hand, the NIPALS algorithm [Wold, 1975] is an iterative robust procedure for solving eigen-values

[1] Multicollinearity refers to a situation in which two or more explanatory variables in a multiple regression model are highly linearly related.

and eigen-vectors problem, see Algorithm 1. Then NIPALS has be used later for PLS solution [Wold, 1975].

Algorithm 1 NIPALS algorithm - Single iteration

Ramdomly initialize $\mathbf{t}$
repeat
  $\mathbf{p} \leftarrow \mathbf{X}^\top \mathbf{t}$
  $\mathbf{t} \leftarrow \mathbf{X}\mathbf{p}$
  $\mathbf{t} \leftarrow \frac{\mathbf{t}}{\|\mathbf{t}\|}$
until Convergence of $\mathbf{t} \leftarrow$ *the resulting* $\mathbf{t}$ *is a single eigen-vector of* $\mathbf{X}$.
$\mathbf{X} \leftarrow \mathbf{X} - \mathbf{t}\mathbf{t}^\top \mathbf{X}\mathbf{y}$ ←*Data deflation*

Henceforward, Lewis proves in [Lewi, 1995] that we can get the same results by using the variance-covariance matrix $\mathbf{X}\mathbf{X}^\top$ instead of $\mathbf{X}$, which is significantly more computationally efficient than NIPALS in the case of dimensionality of the input space exceeds the number of observations. Moreover, he presents NIPALS-PLS algorithm for solving PLS in an iterative efficient way.

Then, Rosipal et al. [Rosipal and Trejo, 2002] used the kernel trick[2] for inducing nonlinear version of the PLS (called KPLS). The KPLS algorithm 2 is based on NIPALS-PLS, however, it uses the kernel form $\mathbf{K} = \Phi(\mathbf{X})\Phi(\mathbf{X})^\top$ instead of $\mathbf{X}\mathbf{X}^\top$.

2.3 Gaussian Processes (GP)

Gaussian Processes (GP) [O'Hagan and Kingman, 1978] introduces probabilistic version of the Representer theorem [Wahba, 1999] through Bayesian modeling. A comprehensive description of GP can be found in Rasmussen and Williams [2006]. GP is the distribution over function space $\mathbf{f}$. It can also be

[2] Proposed in [Aizerman et al., 1964]. the kernel trick is commonly used technique in pattern recognition (*e.g.* KPCA and KSVM).

Algorithm 2 KPLS algorithm

for $i \leftarrow 1 \rightarrow m$ **do**
  Ramdomly initialize u_i
  repeat
    $\mathbf{t}_i \leftarrow \mathbf{K}\mathbf{u}_i$
    $\mathbf{t}_i \leftarrow \frac{\mathbf{t}_i}{\|\mathbf{t}_i\|} \leftarrow$ *normalize vetor* $\mathbf{t}$
    $\mathbf{u}_i \leftarrow \mathbf{y}^\top \mathbf{t}_i$
    $\mathbf{u}_i \leftarrow \frac{\mathbf{u}_i}{\|\mathbf{u}_i\|} \leftarrow$ *normalize vetor* $\mathbf{u}$
  until Convergence in $\mathbf{t}_i$
  $\mathbf{K} \leftarrow (\mathbf{I} - \mathbf{t_i}\mathbf{t_i}^\top)\mathbf{K}(\mathbf{I} - \mathbf{t_i}\mathbf{t_i}^\top) \leftarrow$ *Kernel deflation*
end for
$\mathbf{T} = [\mathbf{t}_1, \cdots, \mathbf{t}_m]$
$\mathbf{U} = [\mathbf{u}_1, \cdots, \mathbf{u}_m]$

viewed as multivariate Gaussian distribution in infinite dimensional space. We use GP to build a nonlinear, probabilistic and non-parametric regression from the input space $\mathbf{X}$ to target space $\mathbf{Y}$.

Parametric modeling

Lets consider a regression function in the form

$$Y = f(X, W) + \epsilon \tag{2.4}$$

where $\epsilon \sim \mathcal{N}(0, \beta^{-1})$, and $f(X, W) = WX$. To find the best mapping parameters $\theta = W, \beta$, we find the Maximum Likelihood estimate of it by $\arg\max_\theta\{L(\theta) = P(Y|X, \theta)\}$. This is called parametric modeling for the regression function. In this form, we build distribution of the parameters θ, so that the mapping function inherits its probabilistic nature from the mapping parameters. Here, we implicitly assume that the prior distribution of the parameters $P(\theta)$ is uniform. However, if we want to encode prior distribution in the parameters space, it is better to use the Bayesian modeling.

Bayesian modleing

In this case, we find the mapping parameterization by maximizing the posterior distribution, which takes the form

$$P(\theta|X,Y) = \frac{P(Y|X,\theta)P(\theta)}{P(Y|X)}$$

where $P(Y|X)$ is the marginal conditional distribution of the data. This inference technique is more robust, since it encodes more information about the distribution of the parameterizations and the data. However, computing these extra terms might be intractable or hard. Here it comes the importance of the Gaussian Processes.

Bayesian nonparametric modeling

GP models the distribution of the mapping function itself without putting any overhead to the mapping parameters. This is done by marginalizing the parameterizations as follows

$$P(y^*|x^*,X,Y) = \int P(y^*|x^*,X,Y,\theta)P(\theta|X,Y)d\theta$$

Moreover, GP defines prior on the mapping function itself as $f \sim \mathcal{GP}(0,K)$, then the value $f(x)$ is a random variable with prior $\mathcal{N}(0,k(x,x))$, where k is a covariance function defined on x-space. Experiments and studies show that the most convenient kernel is the Radial Basis Function (RBF), which takes the form

$$k(x,x') = \beta_1 exp(-\frac{1}{2}(x-x')^\top M(x-x')) + \frac{\delta_{x,x'}}{\beta_2}$$

This form is parameterized in β_1, β_2, M Rasmussen and Williams [2006].

Given a set of training points $\{(x_1, f_1), (x_2, f_2), ..., (x_n, f_n)\}$, we want to evaluate $f(x)$. GP defines the posterior distribution as follows

$$f(x)|X, F \sim \mathcal{N}(\psi^\top K_f^{-1} F, k(x, x) - \psi^\top K_f^{-1} \psi) \tag{2.5}$$

where $X = [x_1, x_2, ..., x_n]^\top$ and $F = [f_1, f_2, ..., f_n]^\top$. $K_f = (K(X, X) - \sigma^2 I)$ is the covariance matrix in the target space and it is defined as a regularized version of the covariance matrix in the input space ($K(X, X)$), and σ is a regularization hyper-parameter. Finally, $\psi(x) = (k(x, x_1), k(x, x_2), ..., k(x, x_n))^\top$ is RBF feature vector of x. The mean of this *Normal* distribution takes the form of regularized least squares of the mapping ($X \to Y$). While the covariance of the point in the function space less than the covariance in the input space by the evidence from the training data, as illustrated in Rasmussen and Williams [2006].

Chapter 3

Supervised Nonlinear Model for Lipreading and Speaker Identification

3.1 Introduction

Audio visual speech recognition (AVSR) has been investigated intensively in the last few decades [Potamianos and Neti, 2004]. Specially after bimodal fusion of audio and visual stimuli in perceiving speech has been demonstrated by the *McGurk* effect [McGurk and MacDonald, 1976]. For example, when the spoken sound /ga/ is seen as /ba/, then most people perceive the sound as /da/ [McGurk and MacDonald, 1976]. Good survey for work on AVSR can be found in [Potamianos and Neti, 2004]. In the last two decades, with the advances in computer vision, visual speech recognition (VSR), also called lipreading, have attracted research attention [Shiell and Terry, 2009]. VSR systems gain importance with the need for controlling machines verbally in a noisy environment. Example of such an environment is the car, where the noise (*e.g.* from motor and radio) makes it very hard for audio speech recognition. Another potential example is to control robot in the outer space where there is no media for audio transmission. Nevertheless, visual speech recognition is a challenging

problem, due to confusion between visemes [1]. Specially, when using information only from plan marker-less and real life images.

Speaker identification and authentication are tightly coupled with speech recognition [Luettin et al., 1996; Sanderson and Paliwal, 2004; Shiell and Terry, 2009]. Speaker identification is defined as the ability to identify the speaker within a group of users from solely speech related features, like voice or mouth motion. Meanwhile, speaker authentication is the ability to authenticate users. We tackle the former problem in this paper. Speaker identification is related to several research fields such as automatic access control, biometrics, and personal privacy issues.

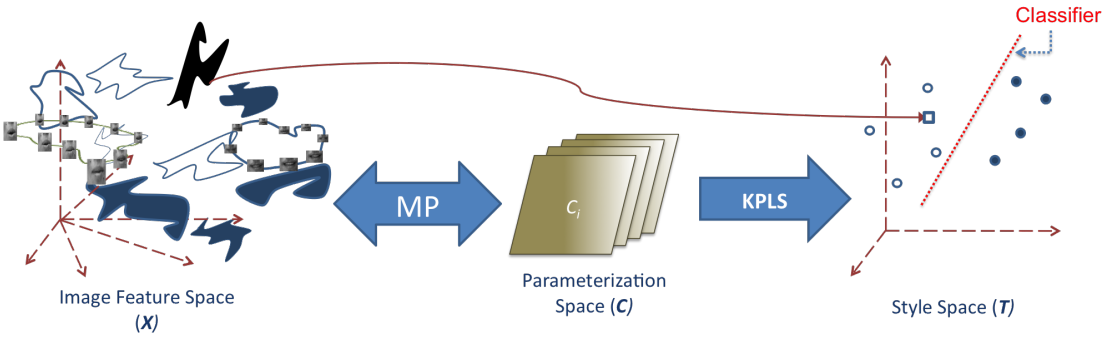


Figure 3.1: This figure embodies the MKPLS framework. It is composed of two phases: Manifold Parameterization (MP) and low-dimensionality embedding using Kernel Partial Least Squares (KPLS). Then, a category based classifier is learned in the KPLS latent space for recognition. In this framework, we learn two latent spaces one for speech recognition and one for speaker identification. Section 3.4 discusses this framework in detail.

In this paper, we present a new approach for embedding of manifolds in a low-dimensional latent space, Figure 3.1. Based on the assumption that frames of each utterance footage lie on smooth low dimensional manifold. We homeomorphic manifold analysis [Elgammal and Lee, 2004a] to parameterize each

[1] Viseme is the visual phoneme. It is defined as the smallest discriminative unit for visual speech

manifold using a nonlinear mapping from a unified manifold representation. However, unlike [Elgammal and Lee, 2004a], where factorization of the manifold parameterization is achieve using unsupervised subspace projection, we factorize the parameterization space in a supervised way. We propose to use kernel partial least square (KPLS) on the mapping coefficient space to achieve a supervised low-dimensional latent space for manifold parameterization. We use two-way projections to achieve two manifold latent spaces, one for the speech content and one for the speaker. The resulting low-dimensional parameterization can be considered as a global spatio-temporal descriptor for each speech sequence, which can be effectively used for speech recognition and speaker identification.

The contribution of the paper can be contrasted in two ways. From learning point of view, we propose a new way to learn a low-dimensional supervised parameterization of manifolds where each manifold is represented as a point in a latent space. From the visual-speech point of view, we propose a new approach for projecting visual speech features into dual latent spaces that are capable of discriminating speech and speaker.

In this work, we use cosine similarity as a kernel on the parameterization space. Moreover, we use two different techniques for classifying new speech clip: one of them is SVM, we learn multi-class SVM based on the projected manifolds. The other one uses KPLS regression for classification on the latent space.

To test the effectiveness of our approach, empirically, we show that our approach outperform previous approaches applied on two databases: AVLetters [Matthews and Cootes, 2002] and OuluVs [Zhao, 2009]. We tackle three different lipreading problems: speaker independent, speaker dependent, and

speaker semi-dependent. In both databases, our approach outperforms for speaker semi-dependent setting by at least 15% over the baseline [Zhao, 2009], and competes in the other two settings.

3.2 Related Work

Several approaches have been adopted for solving the lipreading problem. Two main approaches are commonly used in VSR literature: a Hidden Markov Model (HMM) based approach and classifier based approach. In the HMM approach, after choosing suitable descriptor for the visual unit (usually visemes) corresponding to every node, this descriptor employs as observations for the model. Then HMM model is trained using Baum-Welch algorithm for encoding the stochastic temporal relationship between these observations [Matthews and Cootes, 2002]. Consequently, the Viterbi algorithm [Rabiner, 1989] is used for classification. The classifier based approach is based on extracting a single feature vector for the whole clip of uttered phrase (usually single word, or short sentence), and train a classifier (usually SVM) based on that [Zhao, 2009; Fu and Zhou, 2007]. The proposed approach in this paper belongs to the latter category.

Encoding the dynamics of speech video as a descriptor has a long history within lipreading research. Graphical models have been used extensively in VSR and AVSR. In [Matthews and Cootes, 2002], HMM was used for encoding the visual dynamics of speech using Active Shape Model (ASM) and Active Appearance Model (AAM). A more general Dynamic Bayesian Network(DBN) model has been used in [Saenko and Livescu, 2005] with different visual articulation units called articulatory features. Graph embedding has been used in [Zhou et al., 2011] for estimating the curve that represent the dynamics in

video. These methods try to capture the smooth temporal changes between the used visual units, but they may loose some visual information that may be crucial for discriminating small speech chunks like single letter utterance.

On the other hand, the work in [Zhao, 2009] is based on extracting a single spatio-temporal feature vector for representing the visual and temporal information for the whole speech video. In [Shaikh et al., 2010] optical flow was used for extracting the whole word features. These two approaches outperform in the case of small size videos but it might be sensitive to frame outliers.

In our method, we care about smoothness, since we extract the geometric deformation of the lip-moving manifold and at the same time use all the appearance information for learning a parameterization for this manifold. We test our model on two databases, one contains small clip (AVLetters) and the other database contains slightly longer clips (OuluVs). As the best of our knowledge, we are the first to use homeomorphic manifold analysis and KPLS in the field of visual speech recognition.

3.3 Problem Definition and Framework Overview

We have a set of images sequences representing different activities. Let us denote the k-th sequence by $S_k = \{\mathbf{x}_i^k \in \mathbb{R}^D, i = 1 \cdots n_k\}$, where the images are represented using suitable features of dimensionality D. Let y_k represents the class labels for the k-th sequence. In this paper, for the particular case of speech recognition and speaker identification, $y_k \in \{c_1, \cdots, c_K\} \times \{p_1, \cdots p_L\}$. Here c_i is the activity class label (speech unit), while p_j is the performer class label (speaker). Each sequence lies on a low-dimensional manifold, denoted by $\mathcal{M}_k$, embedded in the feature space $\mathbb{R}^D$. We will denote these manifolds by

instance manifolds. The basic assumption is that all these manifolds are topologically equivalent, however each of them has different geometry in $\mathbb{R}^D$. In other words, all these manifolds are deformed instances of each others. This assumption is fairly met in the domain of activity recognition. For example, periodic locomotive activities intuitively lie on one-dimensional closed manifolds, and hence topologically equivalent. For instance, sequence of features representing a Viseme, starting from a neutral pose and reaching a peak pose, lies on a one-dimensional manifold (curve) in the feature space.

The goal is to achieve a low-dimensional latent space of instance manifolds. In that space each manifold is represented by a single point. Based on that space, instance classification can be achieved. We learn two classification functions $f_{speech}(S)$ and $f_{speaker}(S)$ based on two latent spaces for speech and speaker respectively.

3.4 Manifold KPLS

The dissertation extends Homomorphic manifold analysis (HMA), Section 2.1 as shown in Figure 3.1. This section discusses the proposed framework in details.

3.4.1 Parameterizing the speech manifolds

For computing the manifold parameterization, [Poggio and Girosi, 1990] derived closed form solution for Eq 2.1 as

$$\mathbf{C}_k^\top = (\mathbf{A}_k^\top \mathbf{A}_k + \lambda \mathbf{G})^{-1} \mathbf{A}_k^\top \mathbf{X}_k^\top, \tag{3.1}$$

where $\mathbf{A}_k$ is an $n_k \times n$ matrix with $\mathbf{A}_{(ij)} = K(\mathbf{z}_i, \mathbf{w}_j)$ and $\mathbf{G}$ is an $n \times n$ matrix with $\mathbf{G}_{(ij)} = K(\mathbf{w}_i, \mathbf{w}_j)$. $\mathbf{X}_k$ is the $n_k \times D$ data matrix for instance k. Solution for

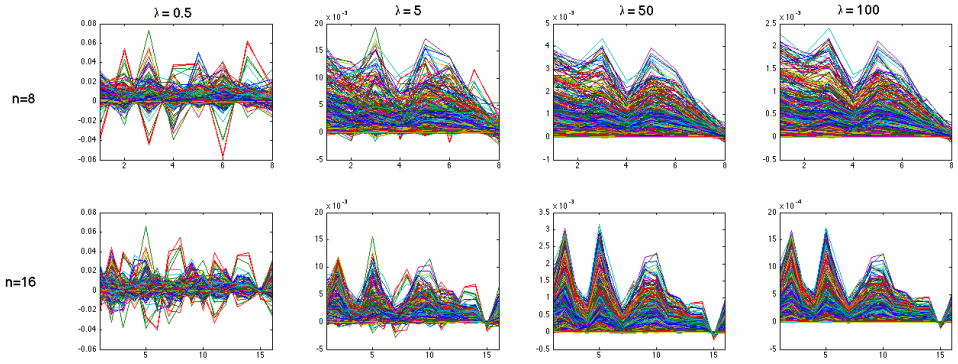


Figure 3.2: The parameterization C is $D \times n$ matrix. Each plot has D lines, and each line is a plot for values progression of a row in C. At large values of λ the parameterization is smooth enough to capture large dynamics in the visual unit.

$\mathbf{C}$ is guaranteed under certain conditions on the basis functions [Poggio and Girosi, 1990]. In this paper, we use Gaussian Radial Basis Function (Gaussian-RBF) for the kernel $K(\cdot, \cdot)$.

The choice of λ and n is crucial for computing the parameterization. Figure 3.2 shows the trade-off between value of λ and n. Acutally, increasing any of them tends to hide fine details and smooth the inferred dynamics curve. This choice depends upon the application. In this work, we need to capture the smooth dynamics in the visual units. Therefore, we choose $\lambda = 50$. It is clear that $n = 16$ expose more variations than with $n = 8$. More information can be useful in some cases and can be more confusing in others. In Section 3.7, the results is reported in terms of both.

3.4.2 Manifold Kernels

Given the manifold parameterization described above, a kernel in the space of manifolds can be defined as a kernel between their parameterizations, *i.e.*

$$K_{manifold}(\mathcal{M}_i, \mathcal{M}_j) \doteq K_{parameterization}(\mathbf{C}_i, \mathbf{C}_j). \tag{3.2}$$

Therefore, we need to define kernels over the space of parameterizations, which consequently, measure the similarity between manifolds in terms of their geometric deformation from the common manifold representation. We can use any valid kernel, in this section we propose using a kernel based on cosine similarity.

In next section, we discuss the discriminant analysis for those parameterizations.

3.4.3 Manifold Latent Embedding

In our framework, we have a set of manifolds represented by $\{(\mathbf{C}_k, y_k), k = 1 \cdots N\}$. y_k is the categorical labeling of the manifold. We need to find nonlinear projection function $\mathcal{F} : \mathbb{C} \rightarrow \mathbb{R}^m$, where $\mathbb{C}$ is the space of all coefficient matrices, and $\mathbb{R}^m$ is a low-dimensional Euclidean space ($m \ll D$), so that $\mathcal{F}$ satisfies the objective

$$\min_{\mathcal{F}} \left\| \mathbf{C} - \mathcal{F}^{-1}(\mathcal{F}(\mathbf{C})) \right\|,$$
$$\max_{\mathcal{F}} cov(\mathcal{F}(\mathcal{C}), \mathbf{y})$$

where $\mathcal{C}$ is the set of parameterizations and $\mathbf{y}$ is the set of responses. We can write $\mathcal{F}$ in a nonlinear regression form as

$$\hat{y} = \Phi(\mathbf{C})\mathbf{B} - \mathbf{E} \tag{3.3}$$

where $\mathbf{B}$, $\mathbf{E}$ are the regression coefficients and residuals respectively.

For solving Eq 3.3, we can use kernel-PCA (KPCR) or kernel-Ridge Regression (KRR). However, using Kernel Partial Least Squares (KPLS) [Rosipal and Trejo, 2002], produces embedding that maximizes the correlation with the response $\mathbf{y}$. The details of KPLS is presented in Section 2.2. KPLS Algorithm 2

finds projection function that embeds the parameterizations $\{\mathbf{C}_k, k = 1 \cdots N\}$ into a low-dimensional latent space $\mathbb{R}^m$, as $\{\mathbf{t}_k \in \mathbb{R}^m, k = 1 \cdots N\}$. The result of KPLS regression is

$$\hat{y} = \mathbf{KU}(\mathbf{T}^\top \mathbf{KU})^{-1}\mathbf{T}^\top \mathbf{y} \tag{3.4}$$

Let $\mathbf{R} = \mathbf{U}(\mathbf{T}^\top \mathbf{KU})^{-1}$. $\mathbf{R}$ works as the projection matrix[Rosipal and Trejo, 2002]. Then, the matrix T, of all embedded points, can be written as

$$\mathbf{T} = \mathbf{KR} \tag{3.5}$$

For a new manifold $\mathcal{M}_\nu$, represented by its parameterization $\mathbf{C}_\nu$ and label y_ν (unknown), the corresponding embedded point can be given by

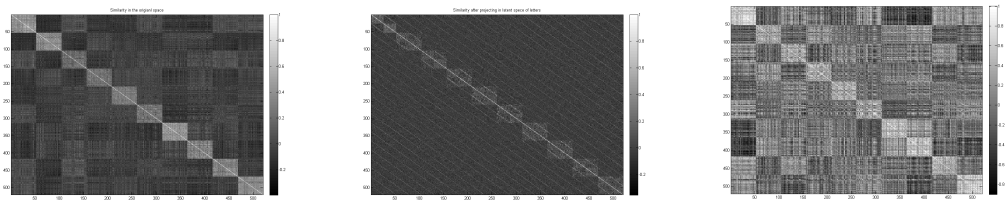
$$\mathbf{t}_\circ = \mathbf{v}_\circ \mathbf{R}. \tag{3.6}$$

Where $\mathbf{v}_\nu = K_{cos}(C_\nu, .)$ (Eq 3.7) is an N-dimensional row vector representing the similarity with all training manifold parameterizations $\{\mathbf{C}_k, k = 1 \cdots N\}$.

3.4.4 Multi-factor Embedding

As aforementioned, we have set of labeled manifold parameterizations $\{(\mathbf{C}_k, y_k);\ k = 1 \cdots N\}$. Consider the case where we have multiple labeling for the same manifold. Therefore, we need to deal with different classification tasks. In this paper, we have two simultaneous tasks: speech recognition and speaker identification.

For phrase/speech recognition, the input manifolds have labeling $y_k^h, k = 1 \cdots N$. We can learn projection matrix $\mathbf{R}^h$ for embedded points $\mathbf{T}^h$ (**A**lgorithm 2). For any new manifold $\mathcal{M}_\nu$, $\mathbf{C}_\nu$ is compute (Eq 3.1), then get the corresponding embedded point by Eq 3.6, as $\mathbf{t}_\nu^h = \mathbf{v}_\nu \mathbf{R}^h$. For speaker identification, we have



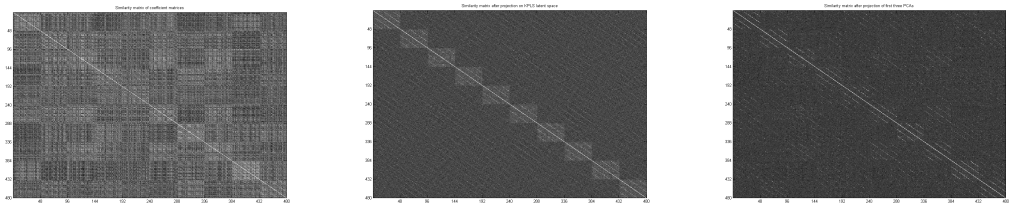
(a) (b) (c)

Figure 3.3: AVletters: similarity among sequences in the manifold parameterization original space 3.3a, and after projection into the KPLS letter's latent space 3.3b, and after projection into the LDA letter's latent space 3.3c.

different labeling $y_k^p, k = 1 \cdots N$. Similarly, we learn the projection matrix $\mathbf{R}^p$ and the embedded points $\mathbf{T}^p$. For new manifold $\mathcal{M}_\nu$, we compute the parameterization $\mathbf{C}_\nu$, then get the corresponding embedded point by $\mathbf{t}_\nu^p = \mathbf{v}_\nu R^p$.

Figure 3.3 shows the affect of projecting into the letters' latent space in the AVLetters database (see Section 3.7.1). In Figure 3.3a , the similarity between speaker dominates the similarity between letters. However in Figure 3.3b , the similarity between letters (represented by diagonals) dominates or at least balances the similarity between speakers. In the same time, self-similarity between speakers still exist which means that the projection preserves the topological relationships in the original space. To compare KPLS against different supervised dimensionality reduction technique such as LDA, Figure 3.3c shows the similarity between the same sequences when LDA used to get the latent embedding. In this figure, we can see that diagonals is barely seen, better than the case in Figure 3.3a, but the effect of speaker's similarity is still dominating the figure.

Similarly, similarity plots are provided in Figure 3.4 between sequences. To make the plots clear in this database we use different settings than mentioned in last section so that



(a) (b) (c)

Figure 3.4: OuluVs: similarity among sequences in the manifold parameterization original space (**a**), and after projection into the KPLS phrase's latent space (**b**), and after projection into the PCA phrase's latent space (**c**).

- We have videos belong to 16 speakers, 10 phrases, three repetitions each.
- The sequences are ordered so that all manifolds belong to the same phrase are consecutive, and every two sequences k_1 and k_2 are two sequences for the same speaker if $|k_1 - k_2| = 16$.
- Consequently, square blocks represent similarity between phrases, while main diagonal and off-diagonals represent similarity between speakers.

The similarity in the original manifold parametrization space Figure 3.4a, in the PCA latent space Figure 3.4c, and in the KPLS latent space Figure 3.4b. Both KPLS and PCA latent spaces have the same dimensionality 100. It is clear that the similarity between phrases (represented by the diagonal blocks) dominates the speaker's similarity only in the KPLS latent space.

3.5 Manifold Classification

At this point, we have a set of labeled low-dimensional representations for manifolds $\{(\mathbf{t}_k, y_k) \in \mathbb{R}^m \times \mathbb{R};\ k = 1 \cdots N\}$. Given a new manifold, parameterized by $\mathbf{C}_v$, we need to classify it, *i.e.* to get its class label $\hat{y}_v$. For achieving this goal, we use two alternative approaches:

Regression for classification (RfC)

Basically, KPLS is regression methodology. So far, we have used only the embedding capability of KPLS. As discussed before, for performing this embedding, we build mapping function from points in Kernel space to points in labels space. Therefore, if we have point in kernel space we can get its corresponding label. To formulate this mapping, we use the following equation which is presented in [Rosipal and Trejo, 2002]

$$\hat{y}_\nu = \mathbf{t}_\nu \mathbf{T}^\top \mathbf{y}$$

where t_ν is computed from Eq 3.6.

Support vector machines (SVM)

Learn one-vs-all SVM classifier for every class on the latent space, and use it for classifying the new embedded point $\mathbf{t}_\nu$, to get $\hat{y}_\nu$.

3.6 Manifold-to-manifold Kernels

The parameterization, extracted out of the first phase of MKPLS, holds the dynamics in each video which encodes speech-related information along with speaker-related information. Because MKPLS uses kernel-based approach for dimensionality reduction, the kernel choice is critical for achieving the best performance. In this section, we investigate several types of kernels.

MKPLS claims that, to define manifold-to-manifold kernel, it suffices to define it in the parameterization space,i.e, $K_{manifold}(\mathcal{M}_i, \mathcal{M}_j) \doteq K(\mathbf{C}_i, \mathbf{C}_j)$. Therefore, we need to define kernels over the space of parameterizations, which consequently, measure the similarity between manifolds in terms of their geometric deformation from the common manifold representation. MKPLS gives us

the ability to plugin any valid kernel. In this section, we investigate several choice of kernels: matrix-based kernels, curve-based kernels and subspace-based kernels.

3.6.1 Matrix-based kernels

Since the dimensionality of all parameterizations is unique, we can measure the similarity between them by measuring the similarity between the corresponding column. This is the idea behind the matrix-based kernels.

Cosine-similarity kernel (Cosine)

We can measure the similarity between columns using cosine the angle between them. As a result, the overall similarity between two parameterizations is the sum over all colmn-wise similarities. Therefore, the cosine-manifold kernel can be defined as

$$K_{\cos}(\mathbf{C}_i, \mathbf{C}_j) = \frac{tr(\mathbf{C}_i \mathbf{C}_j^\top)^2}{||\mathbf{C}_i||_F ||\mathbf{C}_j||_F}, \tag{3.7}$$

where $\|\cdot\|_F$ is matrix Frobenius norm.

Euclidean-distance kernel (Eculid)

In this kernel, we measure the Euclidean distance between the i-th column in parameterization C_1 (u_{1i}) and its corresponding column in parameterization C_2 (u_{2i}). Hence, the overall matrix kernel $\delta = \sum_{i=1}^{n} ||u_i - v_i||_2^2$, and the matrix similarity is

$$K(\mathbf{C}_1, \mathbf{C}_2) = \exp(-\omega\delta) \tag{3.8}$$

where ω is a normalization factor. For $K(.,.)$ to be valid kernel, it needs to be symmetric positive definite matrix (SPD). The exponential function takes care

of the positive definiteness part. For the symmetry, the used distance measure should be metric, which is satisfied for Euclidean distance case.

3.6.2 Curve-based Kernels

In this category, we consider the columns of the parameterization matrix as points in $\mathbb{R}^D$, and the matrix defines a curve connecting those points. The matching between columns should obey the ordering. This means that if two columns i, j from the first matrix match the columns u, v from other matrix respectively, the $u \leq v$ iff $i < j$. For each of the following distances, the parameterization kernel is computed using Eqn 3.8.

Fréchet-distance Kernel (Frechet)

Fréchet distance is a known metric to measure the distance between two curves, that takes into account the location and ordering of the points along the curves. Here, we use discrete Fréchet distance, also known as coupling distance, in which we assume that the curves are piece-wise linear. The basic idea that, each point in one curve is matched with its closest point on the other curve, and the distance d will be the maximum Euclidean distance between each two matched points. At the end, not all points are matched between the two curves.

Edit-distance kernel (EditDist)

The idea of this metric is similar to minimum edit distance between strings. Two main difference between EditDist and Frechet algorithms: in EditDist the overall distance is the sum of distance between all matches while Frechet takes

the maximum of all matches, and EditDist considers unmatched points as being matched with the origin while Frechet ignores the unmatched points. Emperically, we found that the best column-wise distance, in both EditDist and Frechet, is the Euclidean distance.

3.6.3 Subspace-based Kernel

Each parameterization $\mathbf{C}_k$ represents n-dimensional subspace in $\mathbb{R}^D$. Therefore, we can use subspace-to-space metric to measure the similarity in parameterization space. This gives the most general comparison between matrices. Because it considers the subspace spanned by the columns of each parameterization without encoding any ordering.

Grassmannian kernel

Every coefficient matrix $\mathbf{C}_k$ is $D \times d$. Since $D \gg d$, hence $\mathbf{C}_i$ represent d dimensional subspace in $\mathbb{R}^D$. Therefore, the matrix $\mathbf{C}$ belongs to Grassmannian manifold $G_{D,d}$. For more details about Grassmannian manifolds, the reader is referred to Edelman et al. [1998].

There are several approaches for measuring the similarity on Grassmannian manifold, we use the one defined in Harandi and Sanderson [2011].

$$\mathbf{K}_{ij} = a_1 \mathbf{K}_{ij}^{cc} + a_2 \mathbf{K}_{ij}^{proj} \tag{3.9}$$

Where $\mathbf{K}_{ij}^{proj}$, $\mathbf{K}_{ij}^{cc}$ are the projection kernel and the canonical correlation kernel respectively, and a_1, a_2 are weighting constants. The projection kernel is defined by $\mathbf{K}_{ij}^{proj} = \left\| \mathbf{\Pi}_i^\top \mathbf{\Pi}_j \right\|_F^2$ where $\mathbf{\Pi}_k$ is the orthogonal version of the coefficient matrix $\mathbf{C}_k$, computed by Gram-Schmidt orthogonalization algorithm.

The canonical correlation kernel is defined by

$$\mathbf{K}_{ij}^{cc} = \max_{a_p \in span\{\mathbf{\Pi}_i\}} \max_{b_q \in span\{\mathbf{\Pi}_j\}} a_p^\top b_q \tag{3.10}$$

Subject to $a_p^\top a_q = b_p^\top b_q = 1$ if $p = q$, and 0 otherwise. We use two Grassmannian kernels: **Grassm** defined by Eq 3.9 and **GrassmCC** defined by Eq 3.10.

Since Grassmannian distance does not consider the ordering of the parameterization columns, we can encode some temporal information by using the parameterization of difference of the input features. We denote this experiment by **GrassmDiff**.

3.7 Experimental Results

3.7.1 Databases

There are many databases available for AVSR, such as AVLetters [Matthews and Cootes, 2002], AVLetters 2 [Cox et al., 2008], AVICAR [Lee and et al., 2004], AV-TIMIT [Hazen et al., 2004], GUAVE [Patterson and Gurbuz, 2002] and OuluVS [Zhao, 2009]. All AVSR databases can be used for VSR research by simply ignoring the audio information. Our choice is based on several factors. First, we are looking for recent work using solely visual data to compare with. Second, we need to test on different length spoken units. Third, reasonable image resolution. We find that the most adequate databases are AVLetters [Matthews and Cootes, 2002] and OuluVs [Zhao, 2009] for speech recognition and speaker identification. In all experiments, the recognition rate is measured as the ratio between the correctly recognized clips and the total number of clips.

AVLetters database

[2] [Matthews and Cootes, 2002] has ten subjects. Each speaker repeats every English letter ($A \cdots Z$) exactly three times, with a total of 780 video sequences. The speaker was requested to start and end utterance of every letter in a neutral state (mouth closed). No head motion/rotation is allowed from speakers. Every frame is a 60×80 pixel image of the mouth area. This database is very challenging for VSR. The best achieved accuracy for recognizing the spoken letter has been on this database is about 62% [Zhao, 2009]. We use the following setting: For **LBP** features, we tried many configuration. The results is reported in terms of two of them: single cell eight-resolutions ($\mathbf{LBP}_{1:8\times 8}$) and 3×4 cell-grid with four-resolutions (${}_{3\times 4}\mathbf{LBP}^{u_2}_{1:4\times 8}$). For more details about **LBP**, reader is referred to [Ojala, 2002].

OuluVS database

[Zhao, 2009] it consists of ten different everyday phrases. Each phrase is uttered by 20 subjects up to five times. The frame rate was set to 25 fps. The dataset contains sequence of images for mouth area with average resolution of 120×60 pixels. This database is less constrained than AVLetters, so that limited rotation and shift was allowed in the recording time, Figure 3.5(**a**). Not all sequences are perfectly segmented, so that, some sequences have few frames with partial-mouth (Figure 3.5(**b**)) or non-mouth frames (Figure 3.5(**c**)). Some of the outlier sequences (that contain very few mouth/partial-mouth frames) are excluded from the experiment. Consequently, we exclude four speakers with very few sequences remaining (*P004*, *P005*, *P010* and *P016*). The feature

[2] Public version is available on `http://www.ee.surrey.ac.uk/Projects/LILiR/datasets/avletters1/index.html`

(**a**) (**b**) (**c**)

Figure 3.5: OuluVs: (**a**) Regular frames, (**b**) Partial mouth area frames, (**b**) Non-mouth area frames.

configurations used on this database are ($\mathbf{LBP}_{1:8\times8}$) and ($_{1\times2}\mathbf{LBP}^{u_2}_{1:8\times8}$).

3.7.2 Visual speech recognition

We adopt three test protocols for visual speech recognition: speaker independent, speaker dependent and speaker semi-dependent. To present a fair comparison, we restrict ourselves by the configuration specified in [Zhao, 2009].

Speaker Independent VSR (SI):

the challenge here is to recognize the uttered phrase, independent completely of the speaker. By this configuration, we show that our framework generalizes to users is not seen before in the training set. In this experiment, we use one-speaker-out technique.

Speaker Semi-Dependent VSR (SSD):

here we test on one part of the available videos and train based on the remaining set of videos. With one condition that all speakers and phrases have to be presented in the training set. The challenge here is to classify the phrase/expression correctly regardless the user identity.

Table 3.1: Speaker Indepenent - speech recogniton Accuracy on **OuluVs** database

	$n = 8$ and ${}_{1\times2}\mathbf{LBP}^{u2}_{1-8\times8}$					$n = 16$ and ${}_{1\times1}\mathbf{LBP}^{u2}_{1-8\times8}$				
$m =$	10	30	50	80	100	10	30	50	80	100
Cosine	49.22	50.00	51.41	49.84	50.00	50	51.88	49.06	49.06	50.31
Euclid	48.59	55.16	55.78	55.00	55.47	48.44	61.25	58.44	57.50	57.81
EditDist	47.97	50.94	41.25	26.09	21.72	48.44	48.44	43.75	37.81	31.56
Frechet	11.25	13.28	12.50	12.34	13.75					
Grassm	29.84	30.31	32.34	31.09	28.59	22.19	23.75	25.00	19.69	21.25
GrassmCC	29.69	30.31	32.34	31.09	28.75	22.19	23.75	25.00	19.69	21.25

Speaker Dependent VSR (SD):

this experiment tests how far our approach is adequate for use with limited data available. For every speaker, we left one video out for test, and trained based on the remaining videos for the same speaker.

In all configurations, to explore different parameters of MKPLS pipeline, we report empirical comparison for using different kernels described in Section 3.6, along with the manifold parameterization effictive arguments (n and λ) Section 2.1.2. The number (n) of Gaussain-RBF basis ψ that we use to learn the individual manifold parameterization, we show results for $n = 8, 16$. The dimensionlaity of the manifold latent space, we use $m = 10, 30, 50, 80, 100, 130, 200$, which cover wide range of the possible values. Especially for Grassmann-based kernels 3.6, we show the affect of using the parameterization of the LBP of the images itself vs LBP of the images concatenated with parameterization of the discrete difference between those images.

3.7.3 Speaker Independent VSR (SI)

Table 3.1 show the SI speech recognition accuracy for OuluVs for the two configurations.

Table 3.2 and Table 3.3 show the SI speech recognition accuracy for OuluVs and AVLetters, respectively. We can see that for solving speaker independent problem, we need a low-dimensional latent space (about 15 for OuluVs and 25 for AVLetters).

Table 3.2: Subject independent (**SI**) results on **OuluVs** database

	$_{1\times1}\mathbf{LBP}^{u2}_{1-8\times8}$		$_{1\times2}\mathbf{LBP}^{u2}_{1-8\times8}$	
m	**SVM**	**RfC**	**SVM**	**RfC**
10	58.28	55.15	57.18	54.53
15	61.09	62.18	**62.18**	**58.59**
20	60.93	60.46	54.68	57.65
25	**61.56**	**62.34**	56.09	57.50
30	59.06	61.56	55.93	58.28
40	55.62	59.37	56.71	**58.91**
50	58.75	60.46	56.87	58.75

Table 3.3: Subject independent (**SI**) on **AVLetters** database

	$_{3\times4}\mathbf{LBP}^{u2}_{1-3\times8}$		$\mathbf{LBP}^{u2}_{1-8\times8}$	
m	**SVM**	**RfC**	**SVM**	**RfC**
10	32.44	33.46	28.85	29.23
15	38.46	34.87	29.74	32.31
20	41.79	38.85	30.38	33.85
25	**42.69**	**39.87**	28.97	33.59
30	40.77	41.03	31.92	37.82
40	38.33	**42.82**	29.87	**39.36**
50	37.69	41.67	33.08	36.03

3.7.4 Speaker Semi-Dependent VSR (SSD)

Table 3.5 show the SSD speech recognition accuracy for OuluVs database with the two feature configurations. Table 3.4 shows the result of matrix-based kernels and curve-based kernels applied avletters. Table 3.6 and Table 3.7 show

Table 3.4: SSD speech recognition on **AvLetters**

	$n = 8$ and $_{3\times4}\mathbf{LBP}^{u2}_{1:4\times8}$						
$m =$	10	30	50	80	100	130	200
Cosine	50.77	56.67	60.77	62.82	63.85	64.49	63.85
Euclid	51.41	56.41	60.38	64.49	65.13	64.52	64.74
EditDist	51.41	56.54	60.51	64.36	65.13	65.00	64.74
Frechet	23.59	34.62	36.28	34.49	35.64	34.49	29.23

SSD results. In this case, good results need higher dimensional latent space (about 100 for both databases) than in the SI case. This is expected, because

Table 3.5: Subject Semi-dependent speech recognition on **OuluVs** database

	$n = 8$ and ${}_{1\times2}\mathbf{LBP}^{u2}_{1-8\times8}$					$n = 16$ and ${}_{1\times1}\mathbf{LBP}^{u2}_{1-8\times8}$				
$m =$	10	30	50	80	100	10	30	50	80	100
Cosine	62.19	78.13	81.72	81.41	81.72	58.28	77.19	79.22	79.53	79.22
Euclid	61.25	79.06	79.38	79.53	79.84	56.72	75.16	75.00	75.63	75.94
EditDist	62.50	75.63	66.72	43.44	22.81	59.53	70.16	61.25	41.25	35.00
Frechet	29.53	27.81	25.47	17.34	15.97					
Grassm	28.91	37.34	41.87	42.19	39.69	24.38	26.41	29.53	28.44	26.25
GrassmCC	28.91	37.34	41.87	42.19	39.84	24.53	26.41	29.53	28.44	25.94
GrassmDiff						28.13	35.00	37.81	39.17	37.29

in SSD case, almost all variational parameters have been learned already in the training phase, therefore, slightly over-fitting the training data is needed. While in SI case, new variability (*e.g.* new speaker) is presented in testing, therefore, smoothing the projection function is required.

Table 3.6: **OuluVs**:Subject semi-dependent (**SSD**)

	${}_{1\times1}\mathbf{LBP}^{u2}_{1-8\times8}$		${}_{1\times2}\mathbf{LBP}^{u2}_{1-8\times8}$	
m	**SVM**	**RfC**	**SVM**	**RfC**
90	84.68	**83.90**	81.25	**81.56**
100	**84.84**	83.75	**81.87**	81.56
130	84.22	83.75	81.71	81.56
150	84.37	83.75	81.56	81.56
180	84.06	83.75	81.71	81.56
200	84.21	83.75	**82.03**	81.56
220	83.90	83.75	81.40	81.56
250	83.59	83.75	81.71	81.56

Table 3.7: **AVLetters**:Subject semi-dependent (**SSD**)

	${}_{3\times4}\mathbf{LBP}^{u2}_{1-3\times8}$		$\mathbf{LBP}^{u2}_{1-8\times8}$	
m	**SVM**	**RfC**	**SVM**	**RfC**
90	64.10	63.59	63.08	62.56
100	64.23	63.85	62.31	62.18
130	**65.64**	64.87	62.44	61.79
150	65.38	64.49	62.44	61.67
180	65.00	64.10	61.67	61.79
200	64.87	64.10	62.31	61.79
220	65.00	64.10	62.05	61.79
250	64.74	64.10	62.44	61.79

Table 3.8 shows that our framework outperforms the baseline for SSD and compete for SI setting. The third column in Table 3.8a refers to the results of [Zhou et al., 2011], a recent extension to [Zhao, 2009]. The results for [Zhou et al., 2011] are based on what is called *n*ormalized and clean version of OuluVs, while we use the *n*oisy version of OuluVs. Even though, we can compete in

the recognition rate. Moreover, the most practical settings SSD is not presented in this paper. In addition, Figure 3.6 shows more results for OuluVs dataset. Figure 3.6(**a**) shows per-phrase comparison between our results and the results reported in [Zhao, 2009], for SI settings. While Figure 3.6(**b**) shows per-phrase comparison between our framework performance in both SSD and SD settings. For AVLetters database: Table 3.8b shows comparison between our results for AVLetters database and the results in [Zhao, 2009] and [Matthews and Cootes, 2002]. In this dataset, even though the confusion among the letters clips is high, our approach outperform both approaches, specially in the SSD setting.

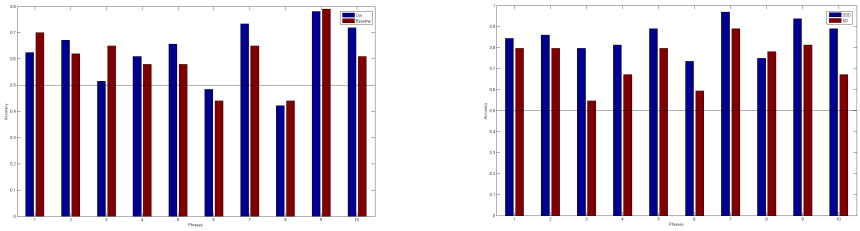
Table 3.8: Comparison with state of the art in configurations: SI, SSD and SD.

(a) **OuluVs**

	Ours	[Zhao, 2009]	[Zhou et al., 2011]
SI	62.34	62.4	**70.6**
SSD	**84.84**	64.2	na
SD	73.59	na	**85.1**

(b) **AVLetters**

	Ours	[Zhao, 2009]	[Matthews and
SI	42.83	**43.46**	n
SSD (third fold)	**64.23**	58.82	57
SSD (total)	**65.26**	62.82	44



(a) (b)

Figure 3.6: On **OuluVs**: (**a**) comparing **SI** results for our approach (blue) and approach used in [Zhao, 2009] (red) . (**b**) comparison between **SSD** results (blue) and **SD** results (red) of our approach.

3.7.5 Speaker recognition

The goal in this experiment is to find the speaker within the register set of users. The challenge is to find the speaker from the limited available information in the mouth area. Moreover, we want to prove that although the manifold parameterization encodes mainly the geometric deformation from the unified manifold to the original data manifold, parameterization also hold speaker-related information. The testing protocol used here is the same as in SSD setting, since we take one repetition out for testing, and we train over all other repetitions.

In both databases, we use the same configuration ($\mathbf{LBP}^{u_2}_{1-8\times 8}$), and the results in both datasets is about 100% regardless of the dimension latent space. That was expected for two reasons: first, we have limited number of speaker (10 in AVLetters and 16 in OuluVs). Second, since we use solely visual information, then the variability due to different speakers is significantly dominating the variability of speech, as shown in Figure 3.3(**a**).

Table 3.9 shows the speaker identification accuracy when applied to OuluVs for the two test configurations.

Table 3.9: Speaker Identification Accuracy on **OuluVs** database

	$n = 8$ and ${}_{1\times 2}\mathbf{LBP}^{u_2}_{1-8\times 8}$					$n = 16$ and ${}_{1\times 1}\mathbf{LBP}^{u_2}_{1-8\times 8}$				
$m =$	10	30	50	80	100	10	30	50	80	100
Cosine	93.91	99.69	99.69	99.69	99.69	92.66	99.69	99.69	99.69	99.69
Euclid	93.75	99.53	99.53	99.53	99.53	93.91	99.53	99.53	99.53	99.69
EditDist	94.22	99.53	99.53	99.37	99.06	92.81	99.69	99.53	99.53	99.53
Frechet	83.91	95.16	89.06	75.94	64.84	88.13	96.09	87.81	62.81	27.66
Grassm	84.69	99.38	99.37	99.53	99.06	92.19	99.22	99.06	98.91	98.44
GrassmCC	84.69	99.38	99.37	99.53	99.06	92.19	99.22	99.06	98.91	98.44

3.8 Conclusion

We proposed a framework that utilized the homeomorphic manifold analysis and KPLS for manifold classification. We tackled two related classification problems speaker identification and speech recognition. We use supervised latent low-dimensional space embedding for solving the simultaneous multi-factor classification problem. We presented three different configurations of lipreading speaker independent, speaker semi-dependent and speaker dependent. The results show that our approach outperform in the semi-dependent setting which we consider the most realistic configuration and perform well in the other two settings.

Chapter 4
Nonlinear Generative Model for Joint Object Recognition and Pose Estimation

4.1 Introduction

Visual object recognition and pose estimation are fundamental problems in the field of computer vision. Recognizing objects and their poses/viewpoints are critical components of vision and robotic systems. With the pervasiveness of robots today, they are required to not only visually recognize objects but also grasp and interact with them. For this reason simultaneously recognizing objects as well as their poses is of utmost importance. The difficulty of the problem lies in the large variation in appearance within object categories and between varying poses of the same objects.

Recent research in the field of generic object categorization and pose estimation can be divided into four tracks, depending on how the models deal with different views and different categories. First, models ignore estimating object pose and learn discriminative object models from training data to categorize objects. The assumption here is that the representation and classifier become view-invariant. With limited size training data, this assumption is hard to meet in reality. Second, learn view-specific models for object category recognition. These approaches discretize the view space into a small number of canonical views (*e.g.* front, right, back, rear-right) and learn classifiers for each of them.

Thirdly, there are approaches that learn category-specific models, with the aim of estimating the viewpoint at a finer granularity, *e.g*. [Torki and Elgammal, 2011; Mei et al., 2011; Schels et al., 2012]. Finally, few recent approaches aim at learning a joint representation of object categories and poses [Lai et al., 2011b; Zhang et al., 2013; Bakry and Elgammal, 2014].

In this work, we tackle the problems of category recognition and pose estimation simultaneously through a joint representation. The intuition behind our model follows from [Zhang et al., 2013], where a common topology is used as a central representation of the multiple views of all objects (*e.g.* a unit-circle manifold for views of an object rotating on a turn-table). All objects are assumed to share the same topology and that there is a *homeomorphism* between these manifolds and the feature/input space for each set of multiple views per object. The space of all mappings (between common topology and input/feature space) for the objects encodes the variation in appearance and shape. The model is then factorized into a bi-linear generative model over two latent factors: *content* and *style*, representing the parameterization on the common topology and parameterization across different mappings, respectively. This model assumes that the points in the mapping space is linearly separable. However it works for the underlined datasets, this assumption is not valid in most cases. On the other hand, the framework in [Zhang et al., 2013] uses sampling for inference. This mandates reducing the dimensionality of the latent space. This lose of information leads to inaccurate pose estimation. In this work, we aim to address the limitations in [Zhang et al., 2013], and leverage the parameterization space.

To achieve this, we extend the Manifold Kernel Partial Least Squares (MK-PLS) framework proposed in [Bakry and Elgammal, 2013] and presented in

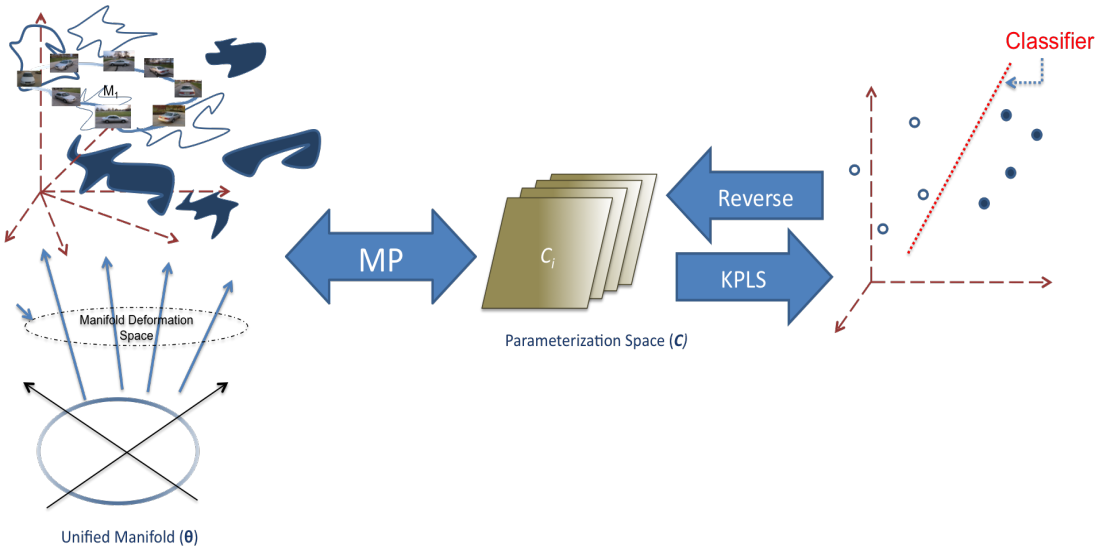


Figure 4.1: Illustration of our dual latent generative model: First, we parameterize the manifold using multiple-view images of a specific object using non-linear RBF mapping. The mapping is from the view-point space (Θ) to the image feature space (**X**). The manifold parameterizations (**C**)are embedded into low-dimensional latent space (**T**), using supervised technique (based on KPLS). Then we learn back map from **T** to **C**. Consequently, each point in the feature space is generated from a point in view-point space and a point in the **T**-space. Meanwhile, we learn style classifier in the latent space. The numbers in this figure shows the order in which the framework training is performed.

Section 3.4, to learn low-dimensional latent space in a supervised way to utilize the available class labels. The framework maximizes the correlation between the points in the latent space and the corresponding labels, see Figure 4.1. In order to support image based inference by propose a generative model that is nonlinear in both category and pose latent variables. In addition, we address the scalability by proposing a hierarchical model where a discriminative model is used to classify super-categories and thus the generative models can be used on these super-categories.

The proposed framework extends Homomorphic Manifold Analysis (HMA) framework in three main dimensions: (1) we propose a novel framework for view-invariant category recognition and pose estimation. This framework presents

a nonlinear method for separating the manifold parameterization (referred to as *style*) and pose variations over the manifold (referred to as *content*) in sets of images. Our generative model is a purely non-linear approach which represents both category and pose using nonlinear latent space embedding. This is in contrast to previous approaches that do not represent the non-linearities across object categories. For this reason our approach has the advantage of being more robust to within-category and pose variations. (2) Our framework of style/content factorization moves the inference of the category and pose from the very high-dimensional feature space into two orthogonal low-dimensional spaces, one for category and the other for pose. Inference in a low-dimensional space guarantees increased accuracy and computational performance. Our framework uses supervised manifold embedding in a low-dimensional space and thus increases the point clustering, and in turn the classification accuracy. (3) We present the use of different distance metrics, different optimization techniques and compare the results of these configurations through extensive experimentation.

4.2 Related Work

Solving object recognition and pose estimation using a manifold-based representation is not novel. The pioneering work of Murase and Nayar explored this idea in [Murase and Nayar., 1995]. However, the recognition in [Murase and Nayar., 1995] is for object instances and not for generic categories. The model used is a linear model based on PCA, while our model is nonlinear in both the pose and the category. Manifold based representations have also been recently used in [Mei et al., 2011], however for object-specific view estimation on video

sequences. We refer the reader to [Savarese and Fei-fei, 2010] for a comprehensive review of recent work on object recognition and pose estimation. We will highlight the most relevant research in this section.

Successful works have been done in estimating the object pose of a known category of objects [Cyr and Kimia, 2004; Mei et al., 2011; Schels et al., 2012; Torki and Elgammal, 2011]. These models have the limitation of being category-specific. This stipulates that independent models are learnt for different object. This does not scale well to many categories with high intra-class variation.

Recently, category recognition and pose estimation have been solved simultaneously (*e.g.* [Savarese and Fei-Fei, 2007; Lai et al., 2011b]). In [Savarese and Fei-Fei, 2007], multiple-view object model is addressed by linking distinct parts of objects from different (discrete) views. This model belongs to the category of limited-pose (discrete-pose) object recognition since it uses a classification approach to deal with pose estimation. Very few works formulate the problem of pose estimation as a regression problem over a continuous space. In [Lai et al., 2011b], a semantic structured tree is built for doing hierarchical inference (object category, instance and pose recognition). This work involves a classification strategy for pose recognition which results in coarse pose estimates and does not fully utilize the information present in the continuous distribution of descriptor spaces. Work presented in [Zhang et al., 2013] and [Torki and Elgammal, 2011] explicitly model the continuous pose variations of objects. In [Zhang et al., 2013], the authors do not solve the problem of large category confusion which we explicitly solve by using our supervised embedding and hierarchical clustering model. In addition we are able to learn non-linearities in the category parameterizations which [Zhang et al., 2013] do not handle. The work in [Bakry and Elgammal, 2014] proposes using feed-forward model

based on object-view manifold to solve object recognition and pose estimation simultaneously. The authors use set of projectors to infer the viewpoint and category. This framework shows superiority in all datasets for recognizing the object and estimating viewpoint. However, inference complexity of this framework increases significantly with the dimensionality of the pose space.

4.3 Dual Latent Generative Model

Images of an *object instance* consists of a set of images of the same object from different viewpoints. Let the k-th instance $S^k = \{(\mathbf{x}_i^k, v_i^k, y^k), i = 1, \cdots, N_k\}$ be the set of images ($\mathbf{x}_i^k \in \mathbb{R}^D$) for the object labeled by its class $y^k \in \{1, \cdots, C\}$. $v_i^k \in \mathbb{R}^v$ is the viewpoint label of the image $\mathbf{x}_i^k$. We propose a generative model that generates points in the image space from two independent latent spaces: pose $\theta \in \mathbb{R}^{d_1}$ and category $\mathbf{t} \in \mathbb{R}^{d_2}$.

The images of each instance are assumed to lie on low-dimensional manifold, which we call *instance manifold* ($\mathcal{M}^k \in \mathbb{R}^D$). For the case of view manifold, we assume that all instance manifolds have the same topology. Under the assumption that all the input object instances are captured using the same degrees of freedom between the camera and object, the corresponding instance manifolds are topologically equivalent but have different geometry in $\mathbb{R}^D$. We can find a unified manifold that is topologically equivalent to all these instance manifolds. The dimensionality of the unified manifold depends on the degrees of freedom allowed for the viewing conditions. In the case of views of an object rotating on a turn-table, and assuming no degeneracy, a viewing circle manifold (one-dimensional unit circle) is used. This can be extended to a viewing sphere (two-dimensional unit sphere). All manifolds are internally parameterized by a latent variable θ lying on the unified manifold, representing the

viewpoint. This unified manifold is *homeomorphic* to the input/feature space of each object instance.

As a result of the homeomorphism, each object's manifold geometry can be parameterized by its geometric deformation from the unified manifold. This parametrization space is view-invariant. The large dimensionality of the manifold parameterization space makes the inference hard and non-robust. Therefore, we learn low-dimensional representation $\mathbf{t}$ for the deformation space. We use supervised kernel-based partial least squares (KPLS Rosipal and Trejo [2002]) to discover this low-dimensional latent space. As a result, the latent variable $\mathbf{t}$ is a view-invariant category representation, and encodes the appearance and geometric characteristics of object instances.

The generative model can be formalized as

$$\hat{\mathbf{x}}(\mathbf{t}, \theta) = \mathcal{A} \times_1 \phi(\mathbf{t}) \times_2 \psi(\theta) \tag{4.1}$$

where ϕ is the nonlinear mapping of the latent space and ψ is a Radial-Basis Function (RBF) over the viewpoints θ[1]. This model is based on nonlinear regression over two factors; the pose θ and latent category representer $\mathbf{t}$. Figure 4.2 shows graphical representation of the proposed generative model. This figure shows that for each point in the image space corresponds to a point on the unified manifold ($\mathbf{\Theta}$) and a point in the low-dimensional parameterization space ($\mathbf{T}$). Moreover, new point in the image space can be generated using the tensor $\mathcal{A}$ in Eq. 4.1.

---

[1] For convenience: we use bold small letters for vectors, bold capital for matrices, calligraphic capital for tensors and regular small for scalars. $\times_i$ denotes mode-i tensor vector multiplication as defined in Lathauwer et al. [2000b].

4.4 Learning the Model

Figure 4.1 visually illustrates the training process of the proposed framework. Each object instance is parameterized using a nonlinear mapping from a unified manifold representation to the input feature space (Section 2.1.2). This manifold parameterization is then embedded in a low-dimensional latent space using a supervised dimensionality reduction (DR) method called KPLS (Section 4.4). We then learn nonlinear mapping from the latent space to the parameterization space to complete the generative model (Section 4.4). The details of each step is provided in this section.

The proposed framework inherits the same pipeline of HMA detailed in Section 2.1. The first step is to find a parameterization for each instance manifold. Let $\{\mathbf{x}_i^k \in \mathbb{R}^D, i = 1, \cdots, n_k\}$ be a set of points on instance manifold $\mathcal{M}^k$. Let $\{\mathbf{z}_i^k \in \mathbb{R}^e, i = 1, \cdots, n_k\}$ be the corresponding points on the unified manifold $\mathbf{\Theta}$.

We learn a regularized mapping functions $\gamma^k(\cdot) : \mathbb{R}^e \rightarrow \mathbb{R}^D$, which maps from $\mathbf{\Theta}$ to each object instance manifold $\mathcal{M}^k$. The mapping can be written in the following matrix form.

$$\gamma^k(\mathbf{z}) = \mathbf{C}^k \psi(\mathbf{z}) \tag{4.2}$$

where $\mathbf{C}^k$ is a $D \times n$ matrix, the vector $\psi(\mathbf{z}) = [k(\mathbf{z}, \mathbf{w}_1), \cdots, k(\mathbf{z}, \mathbf{w}_N)]^\top$ represents a nonlinear kernel map from the embedded representation to a kernel induced space, given a set of RBF centers $\{\mathbf{w}_1, \cdots, \mathbf{w}_N\}$. $k(\cdot, \cdot)$ is an RBF kernel. The solution of Eq 4.2 is shown in Poggio and Girosi [1990] to have a closed form solution:

$$\mathbf{C}^{k^\top} = (\mathbf{A}^{k^\top} \mathbf{A}^k + \lambda \mathbf{G})^{-1} \mathbf{A}^{k\top} \mathbf{X}^{k^\top}, \tag{4.3}$$

where $\mathbf{A}^k$ is a $n_k \times n$ matrix with $\mathbf{A}_{ij} = k(\mathbf{z}_i, \mathbf{w}_j), i = 1, \cdots, n_k, j = 1 \cdots n$ and $\mathbf{G}$ is a $n \times n$ matrix with $\mathbf{G}_{ij} = k(\mathbf{w}_i, \mathbf{w}_j); i, j = 1 \cdots n$. $\mathbf{X}^k$ is the $n_k \times D$ data matrix for manifold instance k.

Supervised Manifold Embedding

The second phase of our framework is to learn a *good*[2] low-dimensional embedding for the parameterizations. Consider Eq 4.2, if we ignore the superscript k, and if we have a test image $\mathbf{x}$, then the objective is to find the best $\mathbf{C}^*$ and $\mathbf{z}^*$ that minimizes the reconstruction error: $\delta(\mathbf{x} - \gamma^k(\mathbf{C}^*, \psi(\mathbf{z})^*))$. However, the generated parameterization ($\mathbf{C}$'s from Eq 4.3) populate a high-dimensional space. This makes solving the objective hard and not robust. Therefore, there is a need to find a suitable low-dimensionallatent space for those parameterizations.

Subspace analysis is used in Elgammal and Lee [2004a] to obtain a latent representation of the manifold parameterization space. However, these approaches do not benefit from available class labels. Alternatively, we propose

using Kernel Partial Least Squares (KPLS) Rosipal and Trejo [2002], detailed in Section 2.2.First, we need to define a kernel on the parameterization space: since $\mathbf{C}$ is $D \times N$ matrix, and $D \gg N$, then $\mathbf{C}$ represents N-dimensional subspace in $\mathbb{R}^D$. Therefore, we can use Cosine-Similarity kernel (CSK) or Grassmannian kernels Harandi and Sanderson [2011]. In this work, we use CSK, for its efficiency, defined by

$$K(\mathbf{C}_i, \mathbf{C}_j) = \frac{tr(\mathbf{C}_i \mathbf{C}_j^\top)^2}{\|\mathbf{C}_i\|_F \|\mathbf{C}_j\|_F}, \tag{4.4}$$

where $\|\cdot\|_F$ is matrix Frobenius norm.

[2] Good in sense that a balance between encoding the spatial relationship between points in the original space and separate points with different labels

Given the instance manifold parameterization $\mathbf{C}^k$ and parameterization kernel and label y^k, we use KPLS for embedding parameterizations space into low-dimensional latent space. The points in this latent space satisfy the two objectives of the PLS.

KPLS maps the point $\mathbf{C}^k$ to latent points $\{\mathbf{t}^k \in \mathbb{R}^m$, for each $k = 1 \cdots N$, by

$$\mathbf{t} = K(\mathbf{C}, \cdot)\mathbf{W} \tag{4.5}$$

where $\mathbf{W}$ is a non-linear projection matrix. Like PCA, the choice of latent dimensionality (m) is a crucial step, since small and large values of m lead to under-fitting and over-fitting of the training data, respectively.

Nonlinear Back Map from Latent Space to Parameterization Space

Almost all linear DR techniques, such as (linear) PLS and PCA, provide 2-way mapping from input space to latent space and vice-versa. In contrast, almost all nonlinear DR techniques do not provide mapping back from latent to input space. KPLS (nonlinear DR) does not provide a closed-form mapping from the latent space $\mathbb{T}$ to the parameterization space $\mathbb{C}$. We need this reverse mapping to complete the generative model in Eq 4.1 and illustrated in Figure 4.1.

Let $\mathbf{c}$ (nD-dimensional vector) is the vectorization of $\mathbf{C}$ ($n \times D$ matrix). Learn non-linear Gaussian RBF mapping $\beta : \mathbb{T} \rightarrow \mathbb{C}$.

$$\mathbf{c} = \beta(t) = \mathbf{A}^\top \phi(\mathbf{t}) \tag{4.6}$$

Where $\phi(\mathbf{t})$ is given by

$$\phi(\mathbf{t}) = K(\mathbf{t}, \cdot) = exp(\sigma \left\| \mathbf{t} - \mathbf{u}_i \right\|) \tag{4.7}$$

where $\mathbf{u}_i; i = 1 \cdots h$ are the centers of the RBF kernel. These centers are computed by K-means algorithm of the points in $\mathbb{C}$. The mapping matrix $\mathbf{A}_{h \times nD}$

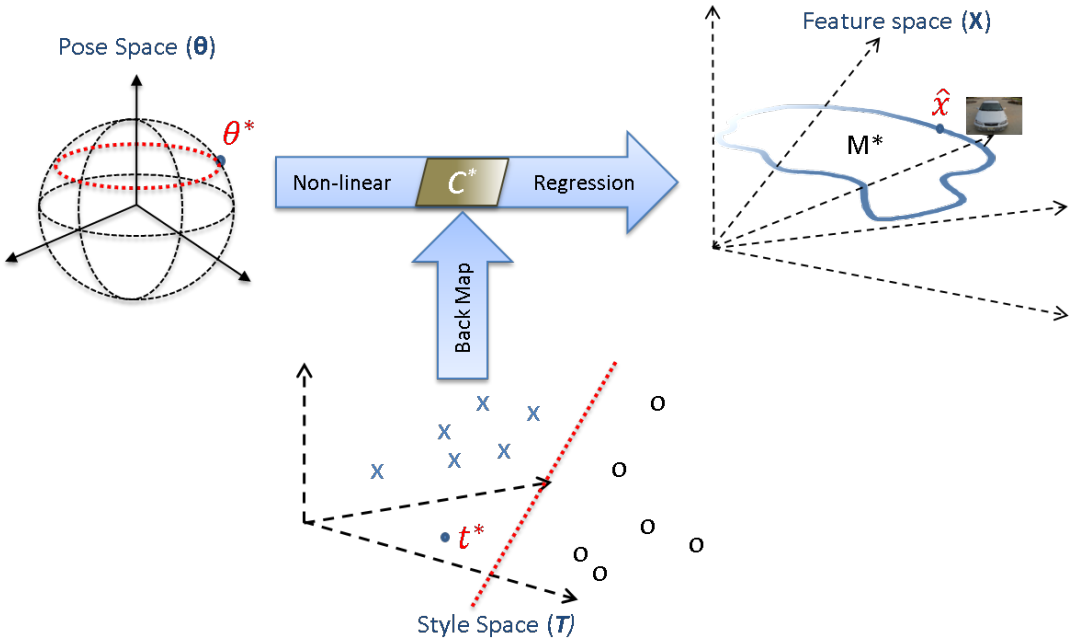


Figure 4.2: Schematic diagram illustrates generating single point in the image feature space ($\hat{x}$) from style estimation (t^*) and pose estimatoin (θ^*) using Eq 4.1. The framework use back-map to regress view-manifold parameterization (C^*) from t^*.

has closed form solution as in Eq 4.3.

By folding **A** and **c**, we get $\mathbf{C} = \mathcal{A} \times_1 \phi(\mathbf{t})$. Where $\mathcal{A}$ is a $h \times n \times D$-tensor. By substituting **C** in Eq 4.2, we get the bi-nonlinear generative model in Eq 4.1.

4.5 Inference

This section shows how to use the proposed generative model for inferring the best pose (v^*) and class (y^*) for a given test image ($\mathbf{x}$). This is done in two steps: first, find the optimal embeddings in the two latent spaces $(\mathbf{t}^*, \theta^*)$ that satisfies the following objective function

$$(\mathbf{t}^*, \theta^*) = \arg\min_{(\mathbf{t},\theta)} \delta(\mathbf{x}, \hat{\mathbf{x}}(\mathbf{t}, \theta)) \tag{4.8}$$

where $\delta(\cdot)$ is a distance metric in the feature space and $\hat{\mathbf{x}}$ is the predicted image. $\hat{\mathbf{x}}$ is computed from Eq 4.1. Figure 4.2 shows visually how the pose space and the style space are composed to generate point in the feature space. Then, process $(\mathbf{t}^*, \theta^*)$ to get the labels (v^*, y^*). In this work, we use two distance metrics to measure the difference between the test image and the predicted image $\delta\{\mathbf{x}, \hat{\mathbf{x}}\}$: Euclidean distance ($\|\mathbf{x} - \hat{\mathbf{x}}\|$) and Normalized Cross-Correlation (NCC) ($1 - \frac{\mathbf{x}^\top \hat{\mathbf{x}}}{\|\mathbf{x}\|\|\hat{\mathbf{x}}\|}$). We adopt two optimization techniques for solving Eq 4.8: gradient-based method and sampling-based methods.

4.5.1 Optimization

Gradient-based

Our gradient-based optimization uses the second order BFGS quasi-Newton optimizer with cubic polynomial line search for optimal step size selection Andrzej P. Ruszczynski [2006].

To use this algorithm, for each value of $\mathbf{t}$ and θ, we need to compute the distance $\delta\{\mathbf{x}, \hat{\mathbf{x}}(\mathbf{t}, \theta)\}$, and its derivative. The derivation is shown in the supplementary material.

Sampling-based

We proposes MCMC sampling Bishop [2006] based algorithm that solve Eq 4.8. We use simulated annealing Korst [1989] to enhance resampling of particles. The details of the algorithm are provided in Algorithm 3.

Algorithm 3 Sampling Algorithm

Initialize particles by the M-means of the available latent points.
repeat
  for Every particle **p** **do**
    Generate N new samples for every old particle **p** - *Use Gaussian as the proposal distribution for generating new samples*
    for $\nu \leftarrow 1 \rightarrow N$ **do**
      $\mathbf{p}_d^\nu \sim \mathcal{N}(\mathbf{p}_d, \sigma) \forall d = 1 \cdots m+1$ *m-D style space T and 1-D pose space θ*
      σ shrinks at further iterations
      $E(\mathbf{p}^\nu) \leftarrow$ distance from Eq 4.8 *Energy value of every particle*
      $A^\nu = E(\mathbf{p})/E(\mathbf{p}^\nu)$ *Acceptance ratio*
      $u \leftarrow \mathbf{Uniform}(0,1)$
      if $A^\nu > u$ **then**
        Accept $\mathbf{p}^\nu$
      else
        Reject $\mathbf{p}^\nu$, and set $\mathbf{p}^\nu \leftarrow \mathbf{p}$
      end if
    end for
  end for
  Re-sample M particles from distribution $e^{-\gamma E(\mathbf{p})}$
until Spatial variance of newly selected particles $\leq \epsilon$

4.5.2 Classification

At this point, for a given test image, we found the best match in both the category latent space ($\mathbf{t}^*$) and the pose latent space (θ^*). We need to infer the label for object category y^* and for pose v^*.

For pose, we have two cases: the actual pose label is discrete, then we use $v^* = k - NN(\theta^*)$, and if the pose label domain is continuous we set $v^* = \theta^*$.

For category, we use SVM in the category latent space, and also we use Regression for classification (RfC), *i.e.* we use the regression results of KPLS Rosipal and Trejo [2002]

$$y^* = {t^*}^\top T^\top \mathbf{y} \tag{4.9}$$

where T is the set of embedded training points in the KPLS latent space (Eq 4.5), and $\mathbf{y}$ is the corresponding labels of the training points.

4.5.3 Hierarchical Model

Dealing with large datasets containing object images with large visual and semantic similarity - *e.g.* tableware, fruits, *etc.*found in the RGBD dataset Lai et al. [2011a] is quite challenging. To solve this we extend our framework to a hierarchical model which performs recursive spectral clustering Duda et al. [2001] to discover clusters of similar objects (as seen in 4.3. For each super-category we use the same generative model described to learn the latent space and perform inference. A Support Vector Machine (SVM) classifier is trained to classify a test image into each super-category and then the generative model for that particular super-category is used to infer the category and pose.

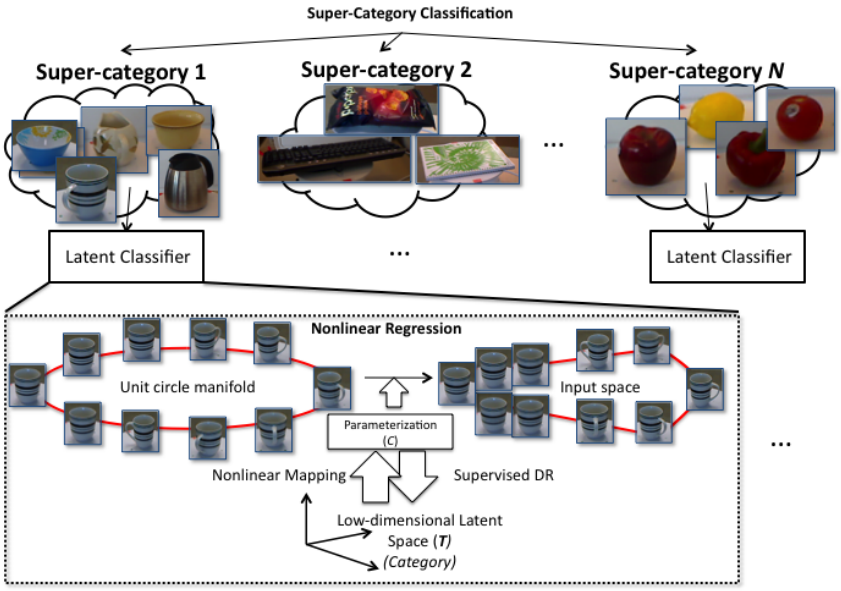


Figure 4.3: The overall hierarchical model. Recursive clustering is used to identify super-categories of similar objects. Our view-invariant latent generative model is then applied to each individual cluster to perform category recognition and pose estimation.

4.6 Experiments and Results

We experiment on three challenging multi-view datasets: 3DObjects [Savarese and Fei-Fei, 2007], RGB-D [Lai et al., 2011a] and EPFL [Ozuysal et al., 2009]. In this section, we outline these experiments.

We use HOG features [Dalal and Triggs, 2005] as our input image representation and we use a unit circle as the unified manifold, i.e, $\theta \in [0, 2\pi]$.

For inference (Section 4.5), we use gradient-based optimization with Euclidean distance and sampling-based optimization with NCC distance metric. The results here are reported based on these best configurations.

4.6.1 3DObjects

We show the experimental results of our work applied to 3D Objects dataset and compare our results to [Savarese and Fei-Fei, 2007; Zhang et al., 2013]. The 3D Objects dataset has images of 10 objects categories. Every category has 10 different instances, differing in brand, color and shape. Every instance has images captured at 3 heights and 3 scales. Every image sequence has 8 poses covering all views: back, back-right, right, front-right, front, front-left, left, back-left.

We show our results for 2 different configurations: 1) 8 classes excluding the farthest scale and 2) All classes (details in supplementary material). Pose accuracy is reported even if the object is incorrectly classified. We use the first configuration for comparison purposes because this is the configuration in prior work ([Savarese and Fei-Fei, 2007],[Zhang et al., 2013]). For all experiment, we train over 7 instances (brands/shapes) and test on remaining 3 brands. The final results are the average of several folds.

	Category				Pose	
	Linear-SVM	SVM	RfC	k-NN	AE_{45°	$AE_{22.5^\circ}$
Gradient(%)	**89.50**	-	**88.65**	**89.68**	**83.25**	**79.19**
Sampling(%)	88.44	85.94	88.31	88.63	80.13	74.94

Table 4.1: Category recognition and pose estimation accuracy (%) on the 3DObjects dataset using our approach. $AE_{22.5^\circ}$ is AE$<22.5^\circ$ and AE_{45° is AE$<45^\circ$

	Sampling	Gradient	[Zhang et al., 2013]	[Savarese and Fei-Fei, 2007]
Average	88.44%	**89.50%**	80.07%	75.65%
Bicycle	99.52%	99.54%	99.79%	81.00%
Car	96.45%	97.22%	99.03%	69.31%
Cellphone	98.10%	**99.54%**	66.74%	76.00%
Iron	88.10%	**90.28%**	75.78%	77.00%
Mouse	50.72%	54.46%	48.60%	86.14%
Shoe	87.56%	**89.35%**	81.70%	62.00%
Stapler	96.08%	**97.63%**	82.66%	77.00%
Toaster	**93.30%**	90.28%	86.24%	74.26%

Table 4.2: Object categorization compared to the baseline. Our framework improves the recognition accuracy for most categories in 3DObject dataset.

Table 4.1 compares the results of two optimization techniques (sampling-based vs gradient-based), against all categorization techniques (Linear-SVM, RBF-SVM, Regression for Classification (*RfC*) and K nearest neighbors (*K-NN*)). The optimization techniques shows similar performance with slight advantage for the gradient-based optimization for both categorization and pose estimation.

Table 4.2 and Table 4.3 show both our sampling and gradient based methods compared against the baselines. We can see an improvement of about 38% in pose recognition (Table 4.3. Table 4.1 shows the comparison of the gradient optimization and sampling optimization in both category and pose recognition using different classification techniques (as described in Section 4.5.2). For pose, we use the same bench mark. $AE_{22.5^\circ}$ is the percentages of images

	Sampling	Gradient	[Savarese and Fei-Fei, 2007]
Average	74.94%	79.19%	57.46%
Front	**75.24%**	74.27%	64.00%
Front-left	66.34%	**73.66%**	40.40%
Left	76.81%	**80.68%**	47.00%
Left-back	76.33%	**88.89%**	62.00%
Back	**80.00%**	78.05%	53.54%
Back-right	73.91%	**84.54%**	71.72%
Right	81.64%	**83.09%**	57.00%
Right-front	67.31%	**70.10%**	64.00%

Table 4.3: 3DObjects dataset: Pose estimation accuracy compared to basedline. Our framework outperformed baseline for most categories.

that have pose estimation error less than 22.5°. Similarly, AE_{45° is the percentages of images that have pose estimation error less than 45°. The gradient-based optimization technique achieves the highest accuracy. Table 4.2 shows the recognition rate of every category compared to the baselines. We used a 13-dimensional KPLS latent space. Inference can be done efficiently on very low-dimensionallatent spaces instead of the very high-dimensionalcoefficient mapping space. From the table, it is clear that our model outperforms state-of-the-art results in most categories. We achieve an overall increase of up to about 10% in accuracy using sampling-based inference and 11.8% using gradient-based inference.

4.6.2 EPFL Cars

To compare with previous work ([Ozuysal et al., 2009],[Teney and Piater, 2013] and [Torki and Elgammal, 2011]), we use the same experiment configuration, we train over the sequences of first ten cars (cars from $1 \rightarrow 10$) and test over the last ten cars (cars from $11 \rightarrow 20$). In this experiment test images are for car instances that are not present in the training data. Our framework finds the

closest instance to the query car instance and estimates the pose.

Table 4.4 shows the results of this configuration compared with previous work. The result are: 90.81% have error less than 22.5 degrees compared to 41.69% reported in [Ozuysal et al., 2009], 70.31% reported in [Torki and Elgammal, 2011] and 78.1% reported in [Teney and Piater, 2013]. Which means that we achieved 16.27% improvement. This shows the power of our framework for continuous pose estimation. We also reduce the mean absolute error (MAE) by about 45%. Leave-one-out has significant improvement over 50% split, the reason behind this is that our algorithm has a wider space (more car instances) to search for the closest point to the query car. More discussion of the results and figures are in the supplementary material.

Method	MAE	$AE_{22.5^\circ}$	AE_{45°
[Ozuysal et al., 2009] - 50% split	46.48	41.69	71.20
[Teney and Piater, 2013] - 50% split	47.40	78.10	79.70
[Torki and Elgammal, 2011] - 50% split	40.60	70.31	80.75
ours - 50% split	**20.35**	**90.81**	**90.81**
[Torki and Elgammal, 2011] - leave one out	35.87	63.73	76.84
ours - leave one out	**11.81**	**95.13**	**95.13**

Table 4.4: Comparison of our results with state-of-the-art baselines on the EPFL multi-view car dataset. For the baselines we report the best results of all configurations

One 50% split may not draw the full picture of the framework performance. More accurate results are shown in Table 4.5. In these two items, we repeated the 50% split experiment, but for four different *fair* splits. One split is trained over cars $1 \rightarrow 10$ and test over the rest. Other split is trained over cars $11 \rightarrow 20$, the third one is trained over the odd numbered cars and the last one trains over the even numbered cars. Figure 4.4 shows $Pr\{AE < x\}$, along with standard deviation across the four splits. This probability is computed in the same way as AE_{x°. So $Pr\{AE < x\}$ is the percentages of images that have pose estimation

Test over	MAE	$AE_{22.5^\circ}$	AE_{45°
$1 \rightarrow 10$	16.35	92.22	92.22
$11 \rightarrow 20$	20.35	90.81	90.81
odd instances	18.34	91.64	91.64
even instances	16.35	92.22	92.22
Average	17.76	91.75	91.75

Table 4.5: Results for our framework on different 50%-splits configurations on the EPFL multi-view car dataset.

error less than x°. Figure 4.5 shows the accuracy on different sequences of the dataset.

4.6.3 RGB-D Dataset

In order to deal with the large number of objects in the RGB-D dataset, we used the hierarchical extension of our model. Referring to fig. 4.3, we can see that similar objects cluster together, *e.g.*, super-category 1 shows round objects, super-category 2 shows flat objects and super-category N shows spherical objects. The accuracy of the top-level SVM classification is 95.6%.

The results are reported in table 4.6. The same accuracy metrics were used as the baseline approaches. We substantially outperform the two baselines in pose estimation over all test images (whether or not they were classified correctly). This is shown in the Avg. Pose (All) column. For the average pose accuracy over the correctly classified images (C), we achieve better accuracy than [Lai et al., 2011b] but slightly less than [Zhang et al., 2013]. For category recognition we achieve less accuracy than the two baselines. For [Lai et al., 2011b] both visual and depth features are used for classification. Our approach only uses visual information. Our model aids pose estimation for finding optimal poses for similar objects even though they are misclassified. For example, similar objects with handles such as mugs and pitchers have similar pose variations

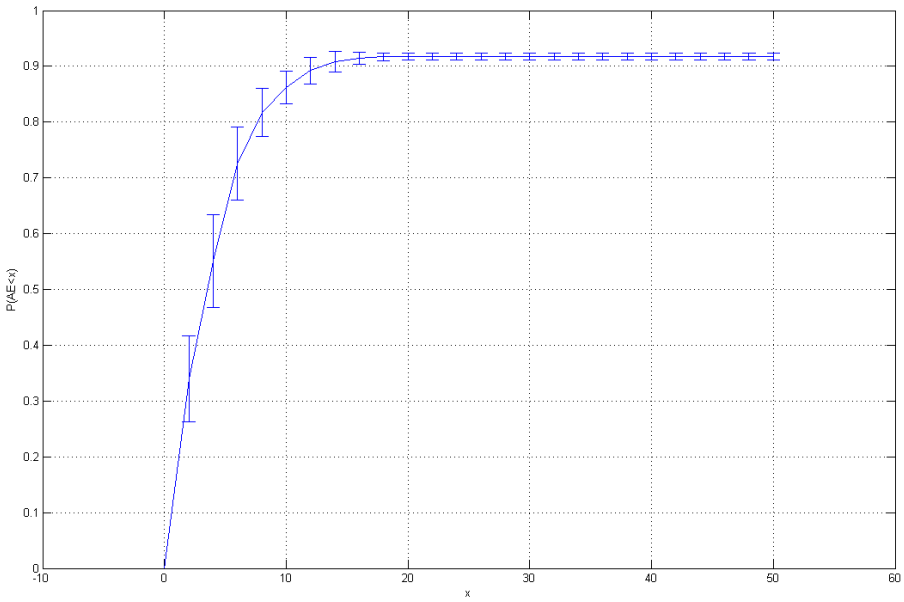


Figure 4.4: In EPFL, pose estimation statistics based on the four 50% splits shown in Table 4.5. The curve represents the probability $Pr\{AE < x\}$ for $x \in [0°, 50°]$, along with standard deviation. Where x is the error value.

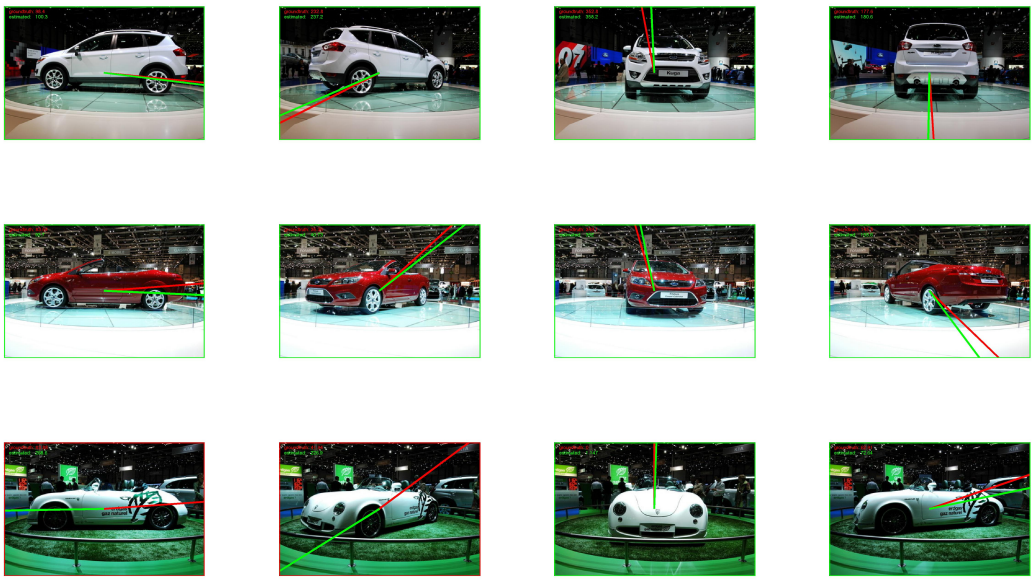


Figure 4.5: Snapshots of the video demo in the supplementary material. Color code: red: ground truth pose, green: estimated pose, frames highlighted green if error less than 22.5° (red otherwise)

Method	Category	Avg. Pose (C)	Avg. Pose (All)
Ours - visual only	85.00	77.31	73.78
[Zhang et al., 2013] - visual only	92.00	80.01	61.57
[Lai et al., 2011b] - visual + depth	94.30	56.80	53.50

Table 4.6: Category recognition and pose estimation accuracy (%) on the RGBD dataset. We report the RGB-only accuracy of [Zhang et al., 2013]. [Lai et al., 2011b] only report their multi-modal RGB+D accuracy.

due to the presence of the handle.

In table 4.6, although the recognition accuracy is less than [Zhang et al., 2013], the overall pose is significantly better. Even with wrong categorization, the method inferred a good representation in the category latent space (which can be a combination of multiple-categories), which enabled the correct pose estimation. The lower recognition accuracy can also be due to the classification scheme used after inference which can be improved.

4.7 Discussion

The proposed model is supervised and nonlinear, in contrast to the linear-unsupervised previous approaches. The use of supervision to achieve the category latent space is fundamental to retain the discriminative ability, while reducing the dimensionality for better inference in this space. This resulted in significantly better results in 3DObjects (category + pose), EPFL-Cars (pose) and RGB-D (pose). For the classification in RGB-D dataset, although the predicted images are notably similar to the corresponding test image, deficit in the recognition accuracy is observed. We believe that this is because KPLS fails to find the best compromise between minimizing reconstruction of the coefficient mapping space and maximizing the correlation with labels (see the supplementary material).

While the use of supervision to achieve latent spaces has been studied before, this has not been addressed in the context of the space of manifold deformations as presented here. Adding supervision is not trivial for high-dimensional parameterization spaces, and results in a nonlinear latent space, which mandates the use of a nonlinear model as we have presented. The nonlinearity in the model requires developing suitable inference methods; we compare between sampling and gradient-based inference methods.

Unfortunately there is no large multi-view dataset to really evaluate scalability. Category-specific models for pose estimation (*e.g.* [Mei et al., 2011; Schels et al., 2012]) are not scalable, since one model per category is needed, and these models do not allow sharing knowledge among similar classes. In contrast, we aim for a model that solves simultaneously for category and pose, where there is one common representation for all categories, hence the scalability potentials. Interestingly, we achieved better pose estimation over many category-specific models, even though our model combines all objects. Compared to [Zhang et al., 2013], our approach is scalable in two fundamental ways. First, in [Zhang et al., 2013], the category latent space was the subspace spanned by all training data, which is still high dimensional for inference. Zhang et al. [2013] used 300 dimensional space for the RGB-D dataset. We only use 20 dimensions and achieve comparable results. In addition, [Zhang et al., 2013] used a k-NN classifier because of the use of the full subspace. This is not scalable and not robust, therefore, supervised DR is needed to achieve a low dimensional representation that retains the discriminative power. Second, we propose a hierarchical Model, which is essential for scalability.

Supervision and nonlinearity are coined together in our use of KPLS. Comparing our sampling results with [Zhang et al., 2013] (also used sampling,

although no algorithm was provided) shows the value of the unsupervised-linear/supervised-nonlinear (88.4% vs 80% Table 4.2).

4.8 Conclusion

We present a novel framework for recognizing object category and pose estimation. The framework uses a generative model that is based on homeomorphic manifold analysis, supervised manifold embedding into a latent space and nonlinear mapping. The advantage of our model is that the inference procedure is moved from the very high-dimensional coefficient mapping space to two low-dimensionalorthogonal pose and category spaces, which makes the inference more accurate and computationally easier. We also incorporate this model into a hierarchical structure to deal with large intra-class variation. We show theoretical basis of our framework and compare our results with the state-of-the-art. We show that our framework both achieves higher performance than state-of-the-art in many configurations and is comparable in others.

Chapter 5
Feedforward Model for Fast Multiview Recognition and Pose Estimation

5.1 Introduction

Visual object recognition is a challenging problem. This is mainly due to the large variations in appearance of objects within a given category, as well as variation of the appearance of an object due to viewpoint, illumination, occlusion, articulation, clutter, *etc*.. Impressive work have been done in the last decade on developing computer vision systems for generic object recognition. Research has spanned a wide spectrum of recognition-related issues, however, the problem of multi-view/view-invariant recognition remains one of the most fundamental challenges to the progress of the computer vision.

The problems of object classification from multi-view setting (view-invariant recognition) and pose recovery are coined together. Inspired by Marr's 3D object-centric doctrine [Marr, 1982], traditional 3D pose estimation algorithms often solved the recognition, detection, and pose estimation problems simultaneously (*e.g*. [Grimson and Lozano-Perez, 1985; Lamdan and Wolfson, 1988; Lowe, 1987; Shimshoni and Ponce, 1997]), through 3D object representations, or through invariants. However, such models were limited in their ability to capture large within-class variability and were mainly focused on recognizing instances of objects. In the last two decades the field has shifted to study 2D

representations, based on local features and parts, which encode the geometry loosely (*e.g*. pictorial structure like methods [Felzenszwalb and Huttenlocher, 2005; Felzenszwalb et al., 2010]) or do not encode the geometry at all (*e.g*. bag of words methods [Willamowski et al., 2004; Sivic et al., 2005].) Encoding the geometry and the constraints imposed by objects' 3D structure are essential for recognition. Most research on generic object recognition bundle all viewpoints of a category into one representation; or learn view-specific classifiers from limited viewpoints, *e.g*. frontal cars, side view cars, rear cars, *etc*..

Recently, there has been an increasing interesest on object categorization from multi-view setting, as well as recovering object pose in 3D. There is a growing interest in developing representations that capture 3D geometric constraints in a flexible way to handle the categorization problem. Savarese and Fei-Fei [Savarese and Fei-Fei, 2007; Savarese and Li, 2008] proposed a part-based model where canonical parts are learned across different views, and a graph representation is used to model the object's canonical parts. Successful recent approaches have proposed learning category-specific detection models that is able to estimate object pose (*e.g*. [Mei et al., 2011; Payet and Todorovic, 2011; Schels et al., 2012; Pepik et al., 2012]). This has an adverse side-effect of not being scalable to a large number of categories while dealing with high within-class variations. Typically literature on this area focus primarily on evaluating the detection, and secondarily on evaluating pose estimation performance, and do not evaluate the categorization performance. In contrast to category-specific representations, in this dissertation we focus on developing a common representation for recognition and pose estimation, which can scale up to deal with a large number of classes.

In this work we consider the problem of modeling the combined object-viewpoint manifold. The shape and appearance of an object in a given image is a function of its category, style within category, viewpoint, and several other factors. Given all these variability collectively, the visual manifold (in any chosen feature representation space) is very hard and even impossible to model. The main goal of this work is to find a computational framework that can untangle such a complex manifold. In particular, we aim at untangling the object-viewpoint manifold, to achieve a model that separates a view-invariant category representation, from category-invariant pose representation.

The proposed framework builds over the model introduced in [Zhang et al., 2013], which mainly proposed to model the category as a "style" variable over the view manifold of objects. This unconventional way is motivated by three observations: 1) the low-dimensionality of the manifold of different views of a given object; 2) the prior knowledge of the view-manifold topology; 3) view manifolds of different objects (under the same view setting) share the same topology (ignoring degeneracy) but differ in their geometry, i.e, view manifolds of different objects are deformed version of each other. In contrast, considering the inter-class and the intra-class variability, even from a give viewpoint, the resulting visual manifold is expected to be quite challenging to model, and can be of infinite dimensions. In [Zhang et al., 2013] a computational framework was introduced that capitalizes on these observations, and models the deformation of different objects' view manifolds. The deformation space is then parameterized to reach a latent view-invariant category space, which is used in recognition. The overall model in [Zhang et al., 2013] is a generative model, where hypotheses about the category and pose were used, within a sampling-based inference approach to minimize the reconstruction

error, given a test image.

There is a mounting evidence of a feedforward computation in the brian for the immediate categorization task [DiCarlo et al., 2012]. This motivated us to seek a forward model, that capitalizes on the same manifold structure observations used in [Zhang et al., 2013], however avoids the challenging inference problem. The sampling-based inference, in [Zhang et al., 2013], constitutes a major limitation to the computational framework. Even though the pose space is very low in dimensionality (one or two depending on the view setting), the view-invariant category latent space is high in dimensionality, which makes sampling not effective, with no guarantee of convergence to the correct answer. In contrast, the current work presents several realizations, which leads to feed-forward computational models that do not require sampling-based inference.

5.2 Framework

This section explains the intuition behind the the proposed framework and introduces the mathematical framework.

5.2.1 Framework Overview

Consider collections of images containing instances of different object classes and different views of each instance. The shape and appearance of an object in a given image is a function of its category, style within category, viewpoint, besides other factors that might be nuisances for recognition. Our discussion do not assume any specific feature representation of the input, we just assume that the images are vectors in some input space. The visual manifold given all these

variability collectively is impossible to model. Let us first simplify the problem. Let us assume that the object is detected in the training images (so there is no 2D translation or in-plane rotation manifold). Let us also assume we are dealing with rigid objects (to be relaxed), and ignore the illumination variations (assume using an illumination invariant feature representation). Basically, we are left with variations due to category, within category, and viewpoint, *i.e.* , we are dealing with a combined *view-object manifold*.

The underlying principle in our framework is that multiple views of an object lie on an intrinsically low-dimensional manifold (*view manifold*) in the input space. The view manifolds of different objects are distributed in that descriptor space. To recover the category and pose of a test image we need to know which manifold this image belongs to, and what is the intrinsic coordinate of that image within that manifold. This basic view of object recognition and pose estimation is not new, and was used in the seminal work of Murse and Nayar [Murase and Nayar., 1995]. In that work, PCA was used to achieve linear dimensionality reduction of the visual data, and the manifolds of different objects were represented as parameterized curves in the embedding space. However, dimensionality reduction techniques, whether linear or nonlinear, will just project the data to a lower dimension, and will not be able to achieve the desired untangled representation.

The main challenge is how to achieve an untangled representation of the visual manifold. The key is to utilize the low-dimensionality and known topology of the view manifold of individual objects. To explain the point, let us consider the simple case where the different views are obtained from a viewing circle, *e.g.* a camera looking at an object on a turntable. The view manifold of each object in this case is a one-dimensional closed manifold embedded in

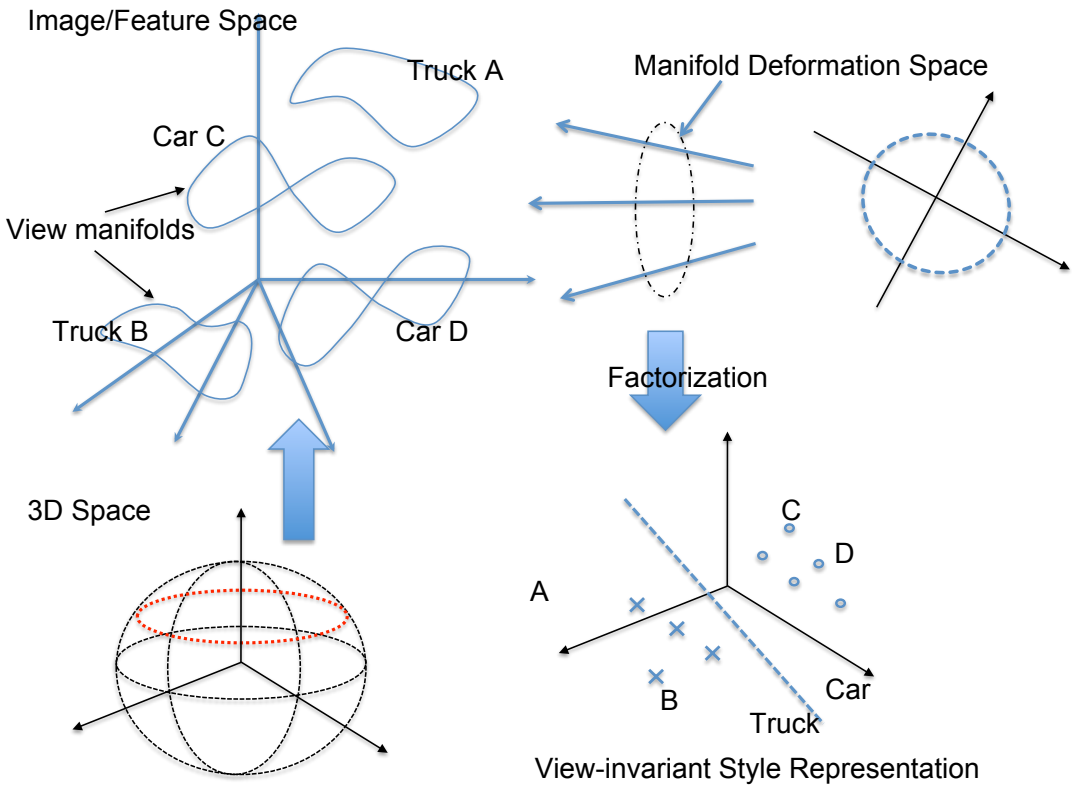


Figure 5.1: Framework for untangling the view-object manifold.

the input space. However, that simple closed curve deforms on the input space as a function of the object geometry and appearance. The visual manifold can be degenerate, for example, imaging a texture-less sphere from different views result in the same image, *i.e.* , the view manifold in this case is degenerate to a single-point.

Ignoring degeneracy, the view manifolds of all objects share the same topology but differ in geometry, and are all homeomorphic to each other. Therefore, capturing and parameterizing the deformation of a given object's view manifold tells us fundamental information about the object category and within category. *The deformation space* of these view manifolds captures a view-invariant signature of objects, analyzing such space provides a novel way to tackle the categorization and within-class parameterization. Therefore, a fundamental

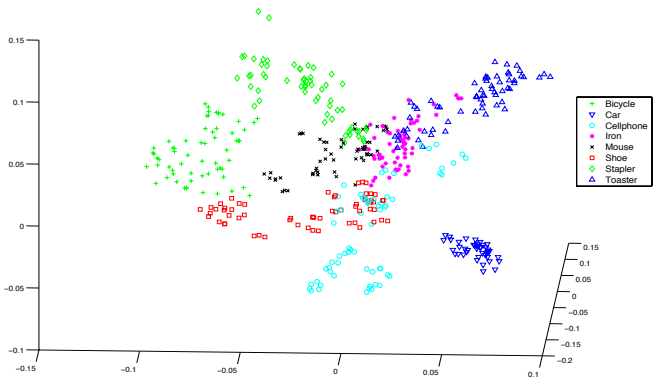


Figure 5.2: Plotting of a three-dimensional unsupervised projection of the view-invariant style parameterization of 473 instances from 3DObjects dataset [Savarese and Fei-Fei, 2007] (obtained from a training set of 3784 images from 8 views). Points of different categories show in different colors and point style. The plot clearly shows the separation between different objects, even in a three-dimensional projection.

aspect in our framework, is that we use the view-manifold deformation as an invariant for categorization and modeling the within-class variations. If the views are obtained from a full or part of the view-sphere around the object, the resulting visual manifold should be a deformed sphere as well. In general, the dimensionality of the view manifold of an object is bounded by the dimensionality of viewing manifold (degrees of freedom imposed by the camera-object relative pose).

5.2.2 Manifold Parameterization

The proposed framework extends Homomorphic Manifold Analysis (HMA) described in Section 2.1. Here, we summarize the mathematical framework proposed in [Zhang et al., 2013], which is the basic for our model, and highlight the challenges. The input are different views of each object instance, where the number views do not have to be same, and the views do not have to be aligned across objects.

Let us denote the view manifold of object instance s in the input space by $\mathcal{D}^s \subset \mathbb{R}^D$ where D is the dimensionality of the input space. Assuming that all

manifolds $\mathcal{D}^s$ are not degenerate (we will discuss this issue shortly), then they are all topologically equivalent, and homeomorphic to each other[1]. Moreover, suppose we can achieve a common view manifold representation across all objects, denoted by $\mathcal{M} \subset \mathbb{R}^e$, in a Euclidean embedding space of dimensionality e. All manifolds $\mathcal{D}^s$ are also homeomorphic to $\mathcal{M}$. In fact all these manifold are homeomorphic to a unit circle in 2D for the case of a viewing circle, and a unit-sphere ($\mathbf{S^2}$) for the case of full view sphere.

We can achieve a parameterization of each manifold deformation by learning object-dependent regularized mapping functions $\gamma_s(\cdot) : \mathbb{R}^e \rightarrow \mathbb{R}^D$ that map from $\mathcal{M}$ to each $\mathcal{D}^s$. Given a Reproducing Kernel Helbert Space (RKHS) of functions and its corresponding kernel $K(\cdot, \cdot)$, from the representer theorem [Kimeldorf and Wahba, 1970; Poggio and Girosi, 1990] it follows that such functions admit a representation in the form

$$\gamma_s(\mathbf{v}) = \boldsymbol{C}^s \cdot \psi(\mathbf{v}) \,, \tag{5.1}$$

where $\boldsymbol{C}^s$ is a $D \times N_\psi$ mapping coefficient matrix, and $\psi(\cdot) : \mathbb{R}^e \rightarrow \mathbb{R}^{N_\psi}$ is a nonlinear kernel map, i.e. $\psi(\mathbf{v}) = [K(\mathbf{v}, \mathbf{v}_1), \cdots, K(\mathbf{v}, \mathbf{v}_{N_\psi})]^T$, defined using a set basis of points $\{\mathbf{v}_i \in \mathbb{R}^e\}_{i=1 \cdots N_\psi}$ on $\mathcal{M}$ (The basis points can be arbitrary and does not need to correspond to actual data points [Poggio and Girosi, 1990]).

In the mapping in Eq. 5.1, the geometric deformation of manifold $\mathcal{D}^s$, from the common manifold $\mathcal{M}$, is encoded in the coefficient matrix $\boldsymbol{C}^s$. Therefore, the space of matrices $\mathbb{C} = \{\boldsymbol{C}^s\}$ encodes the variability between manifolds of different objects, and can be used to parameterize such manifolds. Notice that the dimensionality of these matrices ($D \times N_\psi$) does not depend on the

[1] A function $f : X \rightarrow Y$ between two topological spaces is called a homeomorphism if it is a bijection, continuous, and its inverse is continuous. In our case the existence of the inverse is assumed but not required for computation, *i.e.*, we do not need the inverse for recovering pose. We mainly care about the mapping in a generative manner from $\mathcal{M}$ to $\mathcal{D}^s$.

number of views available each object. We can parameterize the variability across different manifolds in a subspace in the space of coefficient matrices.

Of course the visual manifold can be degenerate or it can be self intersecting, because of the projection from 3D to 2D and lack of visual features, *e.g.* images of a textureless sphere. In such cases the homeomorphic assumption does not hold. The key to tackle this challenge is in learning the mapping in a generative manner from $\mathcal{M}$ to $\mathcal{D}^s$, not in the other direction. By enforcing the known non-degenerate topology on $\mathcal{M}$, the mapping from $\mathcal{M}$ to $\mathcal{D}^s$ still exists, still is a function, and still captures the manifold deformation. In such cases the recovery of object pose might be ambiguous and ill-posed. In fact, such degenerate cases can be detected by rank-analysis of the mapping matrix $\boldsymbol{C}^s$.

The space of manifold deformation functions, encoded by the coefficient matrices $\boldsymbol{C}^s$ is a high-dimensional rich space. Note that all the views of a given object is represented by a single point in that space, parameterizing the geometry of the view manifold of that object, and hence encoding information about its 3D geometry. By projecting the coefficient matrices to a low-dimensional latent space, we can reach a view-invariant representation. Such a representation can be achieved in an unsupervised way or in a supervised way using class labels; in a linear or nonlinear way. In the simplest case, using linear projection, we can achieve a generative model of the data in the form

$$z = \gamma(\mathbf{v}, \mathbf{s}) = \mathcal{A} \times_2 \mathbf{s} \times_3 \psi(\mathbf{v}), \tag{5.2}$$

where $\mathcal{A}$ is a third order tensor of dimensionality $D \times d \times N_\psi$, $\times_i$ is the mode-i tensor product as defined in [Lathauwer et al., 2000a].The variable $\mathbf{v}$ is a representation of the viewpoint that evolves around the common manifold $\mathcal{M}$, which is explicitly modeled. In this model, the variable $\mathbf{s} \in \mathbb{R}^d$ is a parameterization of manifold $\mathcal{D}^s$ that encodes the variation in category/instance of an

object in a view-invariant way. We denote that space by "style" space. Therefore, that space can be used to train category classifiers in a view-invariant way. In this model, both the viewpoint and object/style latent representations, $\mathbf{v}$ and $\mathbf{s}$, are continuous.

Given features from a single test image, denoted by $\mathbf{z}$, recovering the pose and category reduces to an inference problem, where the goal is to find $\mathbf{s}^*$ and viewpoint $\mathbf{v}^*$ that minimize a reconstruction error, i.e.,

$$\arg\min_{\mathbf{s},\mathbf{v}} \|\mathbf{z} - \mathcal{A} \times_2 \mathbf{s} \times_3 \psi(\mathbf{v})\| \tag{5.3}$$

Once $\mathbf{s}^*$ is recovered, a category classifier trained on the style space can be used for categorization. There are different ways to do inference here, for example typical MCMC sampling, or gradient-based optimization can be used.

While the view variable is constrained to a 1D or 2D manifold for the cases of a viewing circle or a viewing sphere, respectively, inference in the style space is very challenging if its dimensionality is high. There is a fundamental trade-off here: Lowering the dimensionality can lead to efficient inference, on the expense of losing the discriminative power of the space; in contrast, keeping the dimensions of the style space high will make the inference unlikely to converge. This is a fundamental limitation of the model, which we try to resolve by avoiding sampling all together, and investigating feed-forward solutions.

5.3 From Inference to Feed Forward

The dissertation proposes a feedforward realization of the model that does not involve inference of the latent variables, yet still capitalizes on the advantages of the model. There are three motivations behind investigating such a feed-forward realization of the model. First, biologically motivated, inspired by

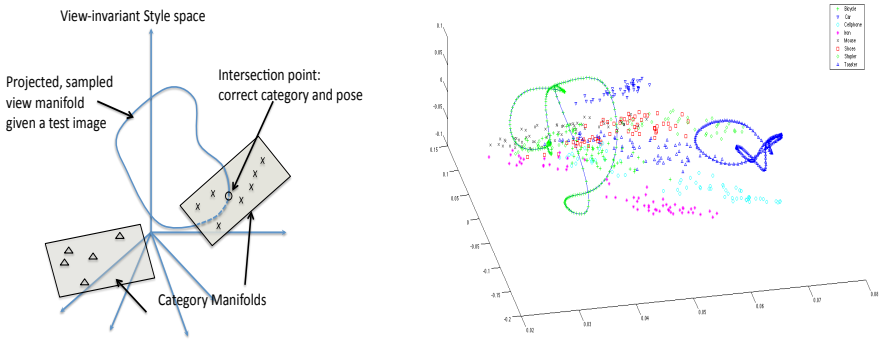


Figure 5.3: Left: Illustration of recovering pose and category by manifold intersection in a view-invariant space. Right: Example of Style-projected Inconsistent View Manifold for two images

the extensive evidence of a cascade of feedforward computation in the brain for solving the immediate categorization problem [DiCarlo and Cox, 2007], we would like to capitalize on the view-invariant property of the style space to achieve a realization of the model that can be implemented in a feedforward manner. Second, computationally, solving the inference problem in Eq 5.3 requires a sampling or a gradient-based search, which might not be desired for real-time applications. Third, from accuracy point of view, there is a tradeoff in choosing the dimensionality of the style space, (recall the style space is a achieved using linear or nonlinear projection of the high-dimensional manifold deformation space). Inference in high-dimensional spaces is notoriously not efficient nor effective. Reducing the dimensionality would lead to efficient inference, on the expense of losing discriminative power in categorization.

View-Invariant Category Manifolds: Let the set of view manifold parameterization matrices be $\{\mathbf{C}^i\}$, where $i = 1, \cdots, M$, is the index of the instances in the training data. Let us assume the case where the factorization in Eq 5.2 is achieved in an unsupervised way, by finding the subspace spanning these matrices. In that case, the factorization is achieve by SVD of the matrix $[\mathbf{c}_1 \cdots \mathbf{c}_M] =$

$\mathbf{U\Sigma V}^\top$, where $\mathbf{c}_i$ is a vectorization of $\mathbf{C}^i$. The columns of $\mathbf{V}^\top$, corresponding to the styles of all training instances. Let us denote these style vectors by $\{\mathbf{s}_i \in \mathbb{R}^d\}_1^M$. Instances of the same category lie on a linear manifold (subspace) in the style space; we call that the *view-invariant category manifold*, and denote it by $\mathcal{C}^k$, where k denotes the category index. Such manifolds capture the within-category variability and also facilitate modeling other variabilities, hence relaxing the rigidity assumption. Figure 5.2 shows an example of the view-invariant space, with different category clearly separated. For the case where no dimensionality reduction take place, *i.e.* $d = M$, the style vectors for the instances of each category would provide orthonormal basis for that category's subspace.

Style-projected Inconsistent View Manifold: The key to achieve a feed-forward realization is, again, in utilizing the low-dimensionality and known topology of the view manifold. Given a test image $\mathbf{z}$ we need to solve the inference problem in Eq 5.3 for the view ($\mathbf{v}$) and style ($\mathbf{s}$) variables. If we know the viewpoint, the problem reduces to solving a least-squares problems for the style variable, which can be achieved by solving the linear system $(\mathcal{A} \times_3 \psi(\mathbf{v}))\mathbf{s} = \mathbf{z}$. Suppose we have a sequence of images of the same object from different viewpoints, $\{\mathbf{z}_i\}_1^n$, and we know the corresponding latent view representation $\{\mathbf{v}_i\}_1^n$, the solutions for the linear system above for every pair $(\mathbf{z}_i, \mathbf{v}_i)$ should all coincide in a single point $\mathbf{s}^*$, since the style-space is view-invariant. However, we only have a single test image, and we do not know the corresponding latent view representation. Instead, if we sample the latent view manifold $\{\hat{\mathbf{v}}_i\}_1^n$ and solve the linear systems $(\mathcal{A} \times_3 \psi(\hat{\mathbf{v}}_\mathbf{i}))\hat{\mathbf{s}}_\mathbf{i} = \mathbf{z}$, we get a sequence of solutions $\{\hat{\mathbf{s}}_i\}$, which constitutes a projection of the view manifold into the style space, using inconsistent pairs $\{(\mathbf{z}, \hat{\mathbf{v}}_i)\}$. Such projection will

also constitute a manifold, we call that *style-projected inconsistent view manifold*, denote it by $\hat{\mathcal{M}}_{\mathbf{z}}$, formally define it as

$$\hat{\mathcal{M}}_{\mathbf{z}} = \{\hat{\mathbf{s}}_i = \mathbf{V}_i^{\dagger}\mathbf{z}\}_1^n$$

where $\mathbf{V}_i = \mathcal{A} \times_3 \psi(\hat{\mathbf{v}}_\mathbf{i})$ is a $d \times D$ matrix, and † denotes the Moore-Penrose pseudoinverse. Note that each image will have its own inconsistent view manifold, hence the use of the subscript. Figure 5.3 shows examples of these manifolds for sample images.

Ideally the correct style $\mathbf{s}^*$ will be a point on that projected view manifold, corresponding to the solution for the pair $(\mathbf{z}, \mathbf{v}^*)$, where $\mathbf{v}^*$ is the closest sampled view to the correct viewpoint. Ideally also the correct style will be the intersection point between $\hat{\mathcal{M}}_{\mathbf{z}}$ and the correct category's manifold $\mathcal{C}^k$. Notice that finding the intersection point directly corresponds to finding the correct viewpoint as well. Figure 5.3 illustrates this process. Realistically, these manifolds might not intersect, especially since we are using sparse sampling of views. Moreover, the category manifolds are hard to model, given the sparse data available at training anyway. Therefore, we need to investigate different ways to achieve an approximate solution. The brute-force method would be a nearest neighbor search between $\{\hat{\mathbf{s}}_i\}_1^n$ and the set of style vectors of the all training instances. Instead we can parameterize $\hat{\mathcal{M}}$ and/or $\mathcal{C}$ and use interpolation to find closest points between them.

Based on the concept explained above, in what follows we propose four different solutions to solve for pose, instance, and category, given image $\mathbf{z}$.

Manifold Intersection: Parametrizing the projected view manifold is easy since its topology and dimensionality is known. The category manifolds are linear in the style space. A simple way to find an approximate solution is to

find the point on $\hat{\mathcal{M}}_{\mathbf{z}}$ closest to each category subspace, This can be achieved by

$$\underset{i,k}{\operatorname{argmin}} \|\mathbf{V}_i^{\dagger}\mathbf{z} - \mathbf{A}_k\mathbf{A}_k^{\top}\mathbf{V}_i^{\dagger}\mathbf{z}\| \tag{5.4}$$

where $\mathbf{A}_k$ is the matrix of orthonormal basis for category k. Unlike the optimization in Eq 5.3, where the search was over continuous spaces for style and view, here the problem reduces to discrete search over categories and sample views. The trade-off in choosing the style dimensionality is no-longer an issue here. The main trade-off here comes from sampling the viewpoint/pose space, however, in most pose estimation applications, only coarse estimation of the viewpoint is needed anyway. However, dense sampling might be necessary to obtain good approximation of the intersection with category manifold, which directly impact the categorization accuracy. This leads to the following three alternative solutions.

View-specific projections: Given a test image $\mathbf{z}$, the correct style $\mathbf{s}^*$ will be a point on the projected view manifold for that image $\hat{\mathcal{M}}_{\mathbf{z}}$, which is most consistent with the correct view $\mathbf{v}^*$, *i.e.* minimizes the reconstruction error. The problem then reduces to minimizing

$$\|\mathbf{z} - \mathcal{A} \times_2 (\mathbf{V}_i^{\dagger}\mathbf{z}) \times_3 \psi(\hat{\mathbf{v}}_i)\|$$

Since $\mathbf{V}_i = \mathcal{A} \times_3 \psi(\hat{\mathbf{v}}_\mathbf{i})$, the above equation reduces to

$$i^* = \underset{i}{\operatorname{argmin}} \|\mathbf{z} - \mathbf{V}_i\mathbf{V}_i^{\dagger}\mathbf{z}\| \equiv \underset{i}{\operatorname{argmax}} \|\mathbf{V}_i\mathbf{V}_i^{\dagger}\mathbf{z}\| \tag{5.5}$$

Basically, this marginalizes the instance/category and provides a way to find the best viewpoint, among the sampled latent viewpoints, that is most consistent with test image. Once the best view, i^*, is found, the style can be directly obtained as $\mathbf{s}^* = \mathbf{V}_{i^*}^{\dagger}\mathbf{z}$. The geometric interpretation of this solution relies on

noticing that the each of the matrices $\mathbf{V}_i\mathbf{V}_i^\dagger$ is an orthogonal projection operator into a view-dependent object-invariant subspace spanned by the columns of $\mathbf{V}_i$. Eq 5.5 is equivalent to finding the view-dependent subspace (spanned by the columns of $\mathbf{V}_i$) where $\mathbf{z}$ is closest to. In that sense, the images in the training data are used to learn these view-dependent object-invariant operators.

One important aspect that we should highlight is that the number of view-specific projector in this model is not restricted by the number of views in the training data. Since manifold parameterization is used to learn the view manifold for each instance, we can sample the view manifold at any arbitrary points $\{\hat{\mathbf{v}}_i\}_1^n$, and hence we can reach any desired number of view-specific projectors.

Instance-specific projections: Using the same rational above, we can also obtain instance-specific view-invariant projectors by marginalizing out the view. Given a test image $\mathbf{z}$, and hypothesizing its corresponding style $\mathbf{s}$, an encoding of the view can be obtained by solving the linear system $(\mathcal{A} \times_2 \mathbf{s})^{\text{-}} = \mathbf{z}$. Recall that $\psi(\mathbf{v})$ is a vector of nonlinear RBF kernels on $\mathbf{v}$, hence we can not obtain $\mathbf{v}$ directly, instead an encoding in an empirical kernel space. Given the set of style vectors $\{\mathbf{s}_i\}_1^M$ obtained from the instances in training data, let us define $D \times N_\psi$ instance-specific matrices $\{\mathbf{B}_i = \mathcal{A} \times_2 \mathbf{s}_i\}_1^M$. The solution for the view representation can be written as $\psi(\mathbf{v}) = \mathbf{B}_i^\dagger \mathbf{z}$. Substituting in the reconstruction error equation, we can reach

$$i^* = \underset{i}{\operatorname{argmin}} \|\mathbf{z} - \mathbf{B}_i\mathbf{B}_i^\dagger\mathbf{z}\| \equiv \underset{i}{\operatorname{argmax}} \|\mathbf{B}_i\mathbf{B}_i^\dagger\mathbf{z}\| \tag{5.6}$$

This marginalizes the viewpoint and provides a set of instance-specific view-invariant orthogonal projectors $\{\mathbf{B}_i\mathbf{B}_i^\dagger\}_1^M$. Eq 5.6 is equivalent to finding the instance-specific view-invariant subspace (spanned by the columns of $\mathbf{B}_i$) where $\mathbf{z}$ is closest to. Once the closest instance subspace is obtained, the pose can be

recovered by finding the closest view in the empirical kernel map space

$$\underset{j}{\operatorname{argmin}} \|\mathbf{B}_{i*}^{\dagger}\mathbf{z} - \psi(\mathbf{v}_j)\| \tag{5.7}$$

Notice that, if the full dimensions of the style space is retained, *i.e.* d=M, the matrices $\mathbf{B}_i$'s reduce to the original coefficient matrices $\mathbf{C}^i$'s. In terms of scalability, the instance-specific solution will not scale well since one projection has to be computed for every instance in the training data, a problem that we will discuss next, to reach category-specific projections

Category-specific projections: The scalability issues highlighted above motivates finding category-specific view-invariant projections, rather than instance-specific ones. The goal is to find a good category representation from the set of matrices $\mathcal{B}_k = \{\mathbf{B}_i | i \in \text{class } k\}$. Equivalently, each of these instance-specific matrices can be represented by an orthonormal basis matrix $\mathbf{U}_i \in \mathbb{R}^D \times N_\psi$. In other words, each instance corresponds to a point on a Grassmann manifold $G(D, N_\psi)$ (the subspace spanned by its column). This put into our disposal all the tools available for Grassmann manifold analysis [Edelman et al., 1998] to obtain a good category-specific representations. For example k-means clustering on Grassmann manifold [Turaga et al., 2011] can be used to achieve a representative category-specific subspace.

Given the set of instance-specific matrices $\mathcal{B}_k$ for the k-th category, we can reach a representation of that category's subspace by merging the subspaces of all its instances. Let $\mathbb{B}_k$ be a $D \times (N_\psi M_k)$ matrix constructed by stacking all the matrices in $\mathcal{B}_k$, where M_k is the number of instances of class k. The column span of this matrix is the union of all the column spans of the instance-specific matrices for this class. Therefore, a category-specific view-invariant projector can be achieved by $\mathbb{B}_k\mathbb{B}_k^{\dagger} = \mathbf{U}_k\mathbf{U}_k^{\top}$, where $\mathbb{B}_k = \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k$ is the truncated SVD

of $\mathbb{B}_k$. Category and pose can be recovered in the same way as in Eq 5.6 and 5.7, by replacing the instance-specific matrices with the category-specific ones.

Discussion: At this point, it is important to contrast the solutions based on the view-specific, instance-specific, and category-specific projections. In terms of scalability, the instance-specific solution will not scale well since one projection has to be computed for every instance in the training data. In contrast the view-specific solution provides a more scalable solution, since the number of views can always be restricted. The view-specific projection also allows the use of discriminative classifiers, *e.g*. SVM in the style space, since it provides a solution for the $\mathbf{s}^*$, in contrast, the instance-specific and the category-specific just find the closest instance or category subspace. Another advantage of the view-specific solution, is that it allows expanding the model to add new objects, even with a single image from a single view point. This can be achieved by computing the corresponding style representation, as mentioned above. A reader might question, why this solution would yield a feedforward computational model. Notice that all projectors are learned offline during training. Finding the best point, whether using nearest neighbor search, or svm classifiers, is also a feedforward computation. Although we do not address detection in this study, it can be achieved through a sliding window approach. However, the challenge is to learn a model for clutter. This can be achieved by projecting clutter training patches using the view-specific projectors, and learning a clutter/object classifier in the style space.

5.4 Experiments

We validated our framework using three multi-view datasets: 3DObjects [Savarese and Fei-Fei, 2007], U-Washigton-RGBD datasets [Lai et al., 2011a], and Multi-View Car Dataset [Ozuysal et al., 2009]. Since we target categorization, instance recognition and pose estimation, in all reported experiments we use ground-truth localizations of objects.

Results on 3DObjects:

3DObjects dataset contains objects from 10 different categories: car, stapler, iron, shoe, monitor, computer mouse, head, bicycle, toaster and cellphone. Each object is imaged from 24 poses on a viewing sphere (8 azimuth angles $\times$ 3 zenith angles), and from 3 scales. We used the entire (all classes) 3DObjects dataset to evaluate the performance of the proposed framework on both object categorization and viewpoint estimation. Similar to [Savarese and Fei-Fei, 2007; Savarese and Li, 2008] we test our model on an 8-category classification task (excluding heads and monitors). However, unlike [Savarese and Fei-Fei, 2007; Savarese and Li, 2008], we do not exclude the farthest scale (which is more challenging). Figure 5.2 shows the learned view-invariant "style" vectors of each object instance, which clearly shows separation between different classes, even in a three-dimensional projection. Because of the limited number of zenith angles (3), we treat each zenith angle as a different viewing circle; *i.e*. all viewing manifolds are considered homeomorphic to a unit circle. To compare to published results, we used a train/test split similar to [Savarese and Fei-Fei, 2007]; we randomly selected 7 object instances out of 10 in each category to build the proposed model, and the rest 3 instances for testing. We

Table 5.1: 3DObjects: Category recognition and pose estimation results (%) for several configurations.

		Categorization Accuracy						Pose Estimation		
		View-specific			Instance-specific	Category-specific	Manifold intersection	View specific	Instance-specific	Manifold intersection
# v	SVM	5NN	7NN	S-Dists						
8		81.86	83.07	79.73	89.65	90.01	76.46	81.86	70.08	63.83
16		82.46	83.74	79.67	89.65	90.01	76.21	80.67	70.08	60.32
20	90.53	82.1	83.34	79.55	89.65	90.01	69.30	80.34	70.08	46.19

used HOG [Dalal and Triggs, 2005] features (20x20x31) as the input space representation. For parameterizing the view manifold, we used 8 RBF centers, (*i.e.* $N_\psi = 8$).

Table 5.1 shows the categorization and pose estimation accuracies using the different setting explained in Sec 5.3. Different rows show the results with different number of sampled views along the view manifold latent space, which is the number of view-specific projectors. For the case of view-specific projectors, after recovering the pose and the style, we evaluated four different classifiers on the style space: one-vs-all linear SVM, 5NN, 7NN, and the distance to the different category subspaces (similar to Eq 5.4 after choosing the best view, *i.e.* minimizing over categories only), denoted as S-Dists. For the view-specific case, the SVM classifier yields the best results. Interestingly, the three types of projectors gave very similar results ($\approx 90\%$). Notice, by construction, that changing the number of sampled views has no effect on the recognition accuracy of the instance-specific or the category-specific projectors. For the pose estimation, we estimate the azimuth angle. Given that the ground truth only has 8 azimuth viewpoints, for the cases where we sample more than 8 views, we approximate the result to the nearest 8 bin case. Not surprisingly, the view-specific projector gave the best results for pose estimation. Overall, the view-specific projector give the best results for both category recognition and pose estimation. Table 5.4-I shows comparison to some of the published

results on this dataset[2].

In a machine with *2.3 GHz Intel Core i7 CPU and 16 GB 1600 MHz DDR3 memory*, each frame of this dataset takes about 4.6 microseconds to be processed (using MATLAB code), excluding the HOG feature extraction, for the instance-specific case.

Results on RGBD:

We evaluated the different setting with the RGB-D dataset [Lai et al., 2011a], which is the largest available multi-view dataset, consisting of 300 instances of 51 tabletop object categories. Each object is rotated on a turn-table and captured using an Xbox Kinect, providing synchronized RGB and depth images. For each object three pitch angles are used: 30,45,60 degrees. Training is done on using 30 and 60 degrees sequences and testing is done on the 45 degree sequences. We use HOG descriptors [Dalal and Triggs, 2005] in both RGB and depth. Unlike the 3DObject dataset, which include completely different objects, the RGB-D is challenging because it has large number of objects, with high appearance similarity among them. Also many objects are almost textureless with symmetric geometry, which makes the pose estimation ill-posed in such cases (*e.g*. an apple or an orange)

Table 5.2 shows the results over different configuration. We use two different setting for manifold parameterization: Setting I uses 11 RBF centers, while Setting II used 20 RBF centers. In both settings we samples 32 viewpoints on the view latent space to generate the view-specific projectors. The description of the different classifiers/metrics is similar to the case of 3D Objects. For the instance-specific projectors we compared two settings: in the first we used the

[2] We mainly compared to approaches that perform categorization and pose estimation. We do not compare to approaches that perform category-pacific detection and pose estimation, since such a comparison will not be fair.

two different heights for each instance to construct a different projector, while in the second setting, we combined the two heights to obtain one instance-specific projector (taking the average of the two style vectors for each instance). We report the instance, category, and pose estimation accuracies. The best results is achieved using the instance-specific projectors.

Table 5.3 summarizes the results, and compares to the state-of-the-art results [Lai et al., 2011b; Zhang et al., 2013]. Comparison to [Zhang et al., 2013] is particularly important since our approach is based on the same formulation. The percentage evaluation metric used is the same as [Lai et al., 2011b]. Following from [Lai et al., 2011b], Average Pose (C) are computed only on test images whose categories were correctly classified. We report the results of our instance-specific projector-II from Table 5.2. We compared the results using different features (RGB and/or Depth). For all feature settings, our instance-specific projector outperforms both [Lai et al., 2011b; Zhang et al., 2013] for instance, category, and pose estimation.

Although our framework is based on [Zhang et al., 2013], and it might be considered as an approximation of it, however we outperforms [Zhang et al., 2013] in all settings. The reason, as we hypothesized in Sec 5.3, is that our approach avoids the sampling-based inference, which has a fundamental dimensionality-accuracy tradeoff, which we do not have. Moreover, our approach is much more efficient. Using Matlab code, on Dell PRECISION 490 with *Intel(R) Xeon (5160@ 3.00GHz 3.00 GHz) CPU - 8 GB memory and 64-bits Windows-7 os* machine (this configuration is far from powerful), we find that the average running time using Instance-Specific approach in this dataset

Table 5.2: RGB-D: Instance, Category, and Pose recognition results (%) using several configurations

	View-Specific					Instance-Specific-I			Instance-Specific-II		
	SVM (classes)		S-Dists (Instances)						(Height-mean)		
Features	Category	Pose	Instance	Category	Pose	Instance	Category	Pose	Instance	Category	Pose
Setting I											
RGB			60.36	76.95	72.51	66.48	85.66	72.24	80.10	94.84	76.63
RGB+D	88.31	73.23	63.80	82.36	73.23	66.19	89.62	71.93	78.63	95.77	75.44
Setting II											
RGB	83.23	72.69	66.24	82.49	74.13	68.24	86.71	73.13			
Depth	51.87	59.02	17.88	39.80	59.02	34.42	71.55	61.30	38.86	76.04	61.65
RGB+D			62.09	82.04	73.36				79.73	96.01	76.01

Table 5.3: Instance and Category recognition, and pose estimation accuracy (%) on the RGBD dataset. Compared to the state of the art [Zhang et al., 2013] and [Lai et al., 2011b].

Method	Instance	Category	Avg. Pose	Avg. Pose (C)
Ours (RGB)	80.10	94.84	76.63	79.78
[Zhang et al., 2013] (RGB)	74.36	92.00	61.59	80.01
Ours (Depth)	38.86	76.04	61.65	70.79
[Zhang et al., 2013] (Depth)	36.18	74.49	26.06	66.36
ours (RGB+Depth)	79.73	96.01	76.01	78.42
[Zhang et al., 2013] (RGB+Depth)	74.79	93.10	61.57	80.01
[Lai et al., 2011b] (RGB+Depth)	78.40	94.30	53.50	56.80

is about 9.2 milliseconds. While the running time of the View-specific approach (with K-NN classifier) is about 0.279 microseconds on the same machine, which shows the power and speed of our framework. This is compared to less than two seconds per frame reported in [Zhang et al., 2013], *i.e.* , our approach much faster and more accurate.

Results on EPFL-CARS:

The Multi-View Car Dataset [Ozuysal et al., 2009], is a challenging dataset, which captures 20 rotating cars in an auto show. It provides finely discretized viewpoint groundtruth, that can be calculated using the time of capturing assuming a constant velocity. Table 5.4-II shows the view estimation results in comparison to the state of the art. All results are generated using view-specific projectors. We build the parameterizations using 15 Gaussian-RBF centers, and the input space is HOG features. We compared the results using 50% splits and leave-one-out splits, which are the typical splits reported in the literature, we

Table 5.4: Categorization and Pose estimation - comparison with state-of-the-art

Table 5.4-I Categorization - 3DObjects

	View-Spec Projectors	Instance-Spec Projectors	Zhang et al [Zhang et al., 2013]	Savarese et al [Savarese and Fei-Fei, 2007]
Average	90.53%	89.56%	80.07%	75.65%
Bicycle	99.54%	99.54%	99.79%	81.00%
Car	99.31%	100.00%	99.03%	69.31%
Cellphone	98.15%	96.29%	66.74%	76.00%
Iron	86.11%	90.74%	75.78%	77.00%
Mouse	52.58%	44.60%	48.60%	86.14%
Shoe	94.07%	92.59%	81.70%	62.00%
Stapler	98.10%	96.21%	82.66%	77.00%
Toaster	98.15%	99.54%	86.24%	74.26%

Table 5.4-II Pose Estimation - Multiview Cars

Method	Split	16 views	8 views
Ozuysal *et al*.[Ozuysal et al., 2009]	50% split	41.69	71.20
Teney and Piater [Teney and Piater, 2013]	50% split	78.10	79.70
Torki and Elgammal [Torki and Elgammal, 2011]	50% split	70.31	80.75
Zhang *et al*. [Zhang et al., 2013]	50% split	87.77	88.48
proposed- 16 views	50% split	**93.94**	**94.13**
proposed- 20 views	50% split	**94.64**	**94.73**
proposed- 32 views	50% split	**94.84**	**94.84**
Torki and Elgammal [Torki and Elgammal, 2011]	leave one out	63.73	76.84
Zhang *et al*. [Zhang et al., 2013]	leave one out	90.34	90.69
proposed -32 views	leave one out	**95.38**	**95.38**

report the average over different splits. More detailed experiments available at the supplementary material.

5.5 Conclusion

We presented a framework for untangling the object-viewpoint visual manifold. We described different approaches based on the framework which learn view-specific object-invariant, instance-specific view-invariant, or category-specific view-invariant projectors from the input space, and described how to solve for the pose and category in each case. Experiment on three multi-view dataset showed the potentials of our proposed approach, we outperform the reported state-of-the-art approaches for recognition and pose estimation on these datasets. Moreover, the approach is shown to be very efficient. The view-specific projectors are the most promising and most scalable approach. We did not target

detection in this work, however, detection can be achieved by running the approach in a sliding window manner, which is a subject of our future research.

Chapter 6

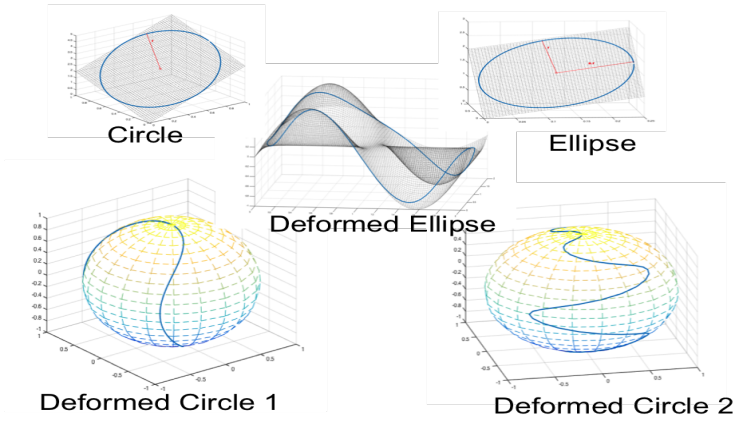
Gaussian Processes for Unified Content Modeling and Style Discrimination

6.1 Introduction

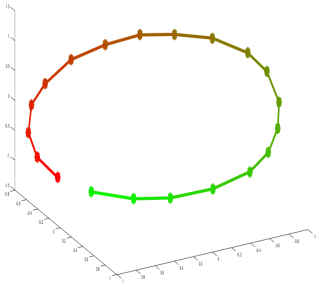
Modeling the latent factors is an essential first step to solve several machine learning problems. Latent factor modeling describes the data distribution in the features space in terms of a set of the affecting parameters. This is achieved by modeling how each parameter changes throughout the data. This process helps to discover the actual manifold span of data, by filling the gaps between the observed points, smooths out the data manifold, and suppresses the observation noise.

Latent factors modeling is challenging because of the existence of nuisance parameters and observation noise. Content and style separation is commonly used to achieve this goal. Content represents the intra-variation while style describes the inter-variations in the data. Yet, it is more challenging to model the variations in multiple point ensembles as function of a single latent factor. This is what we tackle in this study.

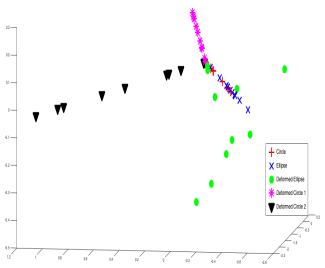
In computer vision particularly, due to the large dimensionality, variability, and noisy nature of the visual data; extracting semantic concepts out of a set of visual observations is challenging. Therefore, many applications in computer vision benefit from separating style variations and learning a unified content



(a) Schematic diagrams for shapes



(b) Content Embedding



(c) Style Embedding

Figure 6.1: Framework overview: **a)** Single dimensional manifolds with different deformations (styles). After applying the proposed framework (GPDS) for style/content separation: **b)** Embedding in content space, **c)** Embedding in style space

manifold such as pose estimation, tracking, facial expression recognition, and others.

To visually illustrate the proposed framework, Figure 6.1 shows the building blocks in the context of a synthetic toy example. This dataset has 50 manifolds with 20 points each. Figure 6.1a shows five instance manifolds, each manifold is a single-dimensional closed curve. This dataset has five different styles (perfect circle, ellipse, deformed ellipse and two more circle styles with different deformations). Every manifold style has multiple instances with different

geometric properties such as radius and orientation. This dataset is built to have two main latent factors: a distinct geometric deformations and a unique intrinsic structure (underlaying topology) of the manifold. Figure 6.1b shows the common topology induced by the proposed framework. While the discriminative geometric features is encoded in the style latent space, Figure 6.1c.

Many machine learning and computer vision applications deal with such structure in the data: Gait recognition, speech recognition, facial expression recognition, human motion tracking, etc. In all these applications, the data can be described in groups of data instances, each instance represents a single video sequence.

For instance in human motion tracking, the data is composed of several instances of motion sequences, and the goal is to predict the future human pose based on set of the previous states. One traditional way to learn this temporal relationship, is to find a single dimensional trajectory that encodes the temporal relation among different human poses. By searching this trajectory for a set of the observed states, we can infer the future pose. For example, Gaussian Process Latent Variable Model (GPLVM) [Lawrence, 2005] and Gaussian Process Dynamical Model (GPDM) [Wang et al., 2005] provides a robust framework to find low-dimensional latent trajectory for a single motion sequence. If we have many instances of motion sequences. These frameworks work well if all the sequences belong to exact the same style. Obviously, this single trajectory is not sufficient to represent different motion styles: different persons and/or different motion classes (walking, jogging, running, jumping, etc.). In this case, we need a parameterized latent trajectory model to accommodate the variations between different styles. To this end, our framework produces a unified content space (motion trajectory) that helps to track the motion, and

utilizes the dynamical variations to find more robust style representations as well.

In general, the proposed framework is designated to deal with any dataset that composed of disjoint sets of instances, and points in each instance lie on a low-dimensional manifold that lives in arbitrary dimensional feature space. In other word, the data lies on multiple instances of manifolds sharing the same topology but differ in the geometry. Our goal is to find the hidden topology of the underlaying manifolds, and simultaneously use it to isolate the intrinsic structure (the manifold deformation) of the instance manifolds (content) from the inter-instances variations (style).

To achieve this goal, we derive a closed form for the likelihood of the points in the feature space as a function of two orthogonal latent spaces. By maximizing this joint probabilistic objective function, we can infer the embedding of the observation points into the latent spaces, see Section 6.4.

For training the model, we derive an approximation algorithm that optimizes the joint likelihood by sequentially optimizing two separate GP mappings (content-GP and style-GP). Starting by arbitrary random latent initialization, the proposed approach sequentially uses Gaussian Process Latent Variable Model (GPLVM) [Lawrence, 2005] to find the best embedding in every latent space separately, see Section 6.5.

Section 6.2 puts the contributions in the context of the related work. Section 6.3 states formal definition for the problem with the essential background to understand the proposed framework. In Section 6.7, we show the efficiency of our framework by empirical experiments on synthetic and motion dataset (CMU-MOCAP [Leordeanu and Hebert, 2005]), and speech data dataset (AVLetters [Matthews and Cootes, 2002]) and HumanEva-I [Sigal and Black, 2006].

6.2 Related work

Modeling the interaction between latent factors has attracted many researchers in the last decade. Bilinear model for separating style and content factors in the data has been proposed in [Tenenbaum and Freeman, 2000]. This was applied for separating pose variations from the person identity. Instead of extracting single content and single style, The authors in [De Lathauwer et al., 2000] porposed multi-linear singular values decomposition (HOSVD), which has been employed by many researchers to extract multiple factors [Vasilescu and Terzopoulos, 2003]. Nonlinear factorization is intended to uncouple more complex relation between latent factors. It has been used for separating the style of motion manifold as deformations of common content manifold [Elgammal and Lee, 2004a].

Gaussian Processes (GP) [O'Hagan and Kingman, 1978; Rasmussen and Williams, 2006] introduces probabilistic version of the Representer theorem [Wahba, 1999] through Bayesian modeling. It generalizes the Gaussian probabilistic inference for functional modeling. Gaussian Process latent variable model (GPLVM) [Lawrence, 2005], uses the power of GP for performing a robust nonlinear dimensionality reduction. GPLVM find the locations in the latent space that maximize the likelihood of the mapping from the latent space to the feature space.

Gaussian Process Dynamical Model (GPDM) [Wang et al., 2005] uses GPLVM for latent embedding and uses nonlinear auto-regressive model as prior to encode the dynamics and smoothness in the latent space. However, for multiple sequences, GPDM produces significantly different embeddings even for sequences of similar styles such as walking. This limits the robustness for any

further inference based on the the generated embedding. Therefore, Balanced-GPDM [Urtasun et al., 2006] adds more priors regularize embedding the manifolds. This helps the authors to use the mean embedding trajectory to track the human motion [Urtasun et al., 2006], and infer the body pose. In this context, our work is the first to learn a unified latent embedding from multiple sequences. This guarantees robust back mapping into the observation space.

In multi-factor GPDM (MGP) [Wang et al., 2007], the authors model human locomotion by introducing a product space of all latent factors (motion state, subject and gait type). They used GP for mapping the points in the product space to the points in the observation space. They showed that the kernel of this mapping is Hadamard product of the factor's kernel. In contrast, we generate orthogonal embeddings into separate content space and style space(s). This separation allows for more efficient model learning and more robust inference.

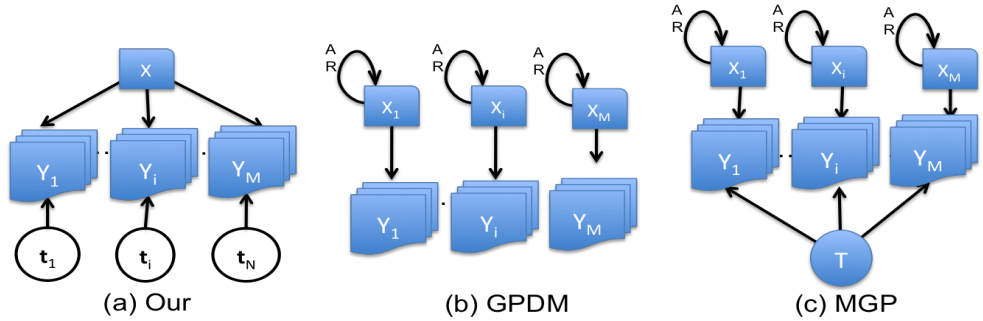


Figure 6.2: Schematic diagram comparison of the three frameworks **a)** Ours, **b)** GPDM [Wang et al., 2005] and **c)** Multifactor GPDM [Wang et al., 2007]. Blue shape is for set of points, white means single point. Square and circle is for content and style embedding respectively.

Figure 6.2 contrast the difference between our proposed framework with GPDM [Wang et al., 2005] and MGP [Wang et al., 2007]. In this figure, Y represents a single instance manifold in the feature space, X and T represent its content and style embeddings respectively. GPDM gives independent content

embedding (X_i) for every sequence (Y_i). In our framework, we find a unified content embedding and a single style space for all sequences $\{Y_i\}_{i=1}^{N}$.

On the other hand, MGP finds embedding X_i in the latent space for each sequence Y_i and the sequences are related with set of point in the style space (T). Number of styles should be know in advance, otherwise every sequence is allocated different point. Compared to Eq 6.5 in our framework, MGP can be described by $y(x) = \sum_{j=1}^{S} t_j \mathbf{w}_j \phi(x)$. In contrast, our framework finds a unified latent trajectory (X) in the content space for all sequences. The mapping from X to every sequence Y_i is parameterized by point in the style space t_i. Unifying the content embedding is more reasonable for two reasons: The content is usually parametrized in global factor such as time, however style is application dependent. We usually need to encode common properties in the content space not in the style space.

Analyzing multiple sources of information, such as embedding a set of motion sequences, can be though as multi-view learning. In Shared-GPLVM [Ek et al., 2008b] and Joint-GPLVM [Navaratnam et al., 2007], the authors were looking for common latent embeddings. However, counting only the similarity and ignoring the dissimilarity leads to noisy embedding. Therefore, some researches counted for both a shared embedding and a private embedding [Ek et al., 2008a; Salzmann et al., 2010; Damianou et al., 2012]. Shared embedding models the common variational attributes among the input set of sequences, while the private part separates out all the private variations for each view in separate space. Our framework is related to this track by finding a common latent content space. In contrast, our framework uses a single continuous space to model the private variability between the sequences.

6.3 Problem Definition

The target dataset has multiple disjoint set of instances. The points in each instance are assumed to lie on a low-dimensional manifold, and all instances share a common topology, but differ in the geometry, as shown in Fig 6.1a. Our goal is to find embedding of these data in separate content and style latent spaces, as well as probabilistic non-linear function that generates points in the observation space in terms of the inferred embeddings. In this work, the content is low-dimensional latent manifold that encodes the underlying topology of all instances. In the following derivation, we assume a single style factors. This restriction will be relaxed later on, to allow multiple style embeddings corresponding to several latent factors.

Let the instance manifold s defined by ($\mathcal{M}_s = \{y_1^{(s)}, y_2^{(s)}, ..., y_N^{(s)}\}$), where $y_i^{(s)} \in \mathbb{R}^D$, $1 \leq i \leq N$ is a single point in the feature space. Let $Y_s = [y_1^{(s)}, y_2^{(s)}, ..., y_N^{(s)}]^\top$ is stacking for the points in $\mathcal{M}_s$. We are looking for unified latent embedding $\mathcal{U} = \{x_1, x_2, ..., x_N\}$, so that $y_i^s = f^s(x_i)$, $x_i \in \mathbb{R}^{d_x}$; $\forall i = 1...N$. Let $X = [x_1, x_2, ..., x_N]^\top$ be the stacking of the points in $\mathcal{U}$.

6.4 The Proposed Approach

Assuming that the latent embedding lies on a unified content manifold $\mathcal{U}$, we are looking for mapping functions $f^s : \mathcal{U} \rightarrow \mathcal{M}_s$; $\forall s \in \{1, ..., M\}$. In our work, we use Gaussian process (GP) to model this mappings ,*i.e,*

$$Y_s \sim \mathcal{GP}(\mathbf{0}, K_X)$$

where $K_X = k_x(x_i, x_j); \forall\ x_i, x_j \in \mathcal{U}$, and k_x is a covariance function defined on $\mathbb{R}^{d_x}$.

For every observation point $y^* \in \mathcal{M}_s$, the embedding point $x^* \in \mathcal{U}$ is maximum a posteriori (MAP) estimate [Rasmussen and Williams, 2006]

$$x^* = \arg\max_x P(y^*|x, Y_s, \mathcal{U})$$

and

$$y^*|x^*, Y_s \sim \mathcal{N}(\psi_*^\top \hat{K}_X^{-1} Y_s, k_x(x^*, x^*) - \psi_*^\top \hat{K}_X^{-1} \psi_*) \tag{6.1}$$

where $\psi_* = \psi(x^*)$ is RBF feature vector of x^* and $\psi(.) = [k_X(., x_1), k_X(., x_2), ..., k_X(., x_n)]^\top$, $\hat{K}_X = (K_X + \lambda_X I)$, λ_X regularizes the mapping and I is the identity matrix.

To find unified content embedding for all instance, we parameterize this mapping from the content manifold $\mathcal{U}$ and each of the instance manifolds $\mathcal{M}^s$ ($f^s : \mathcal{U} \rightarrow \mathcal{M}^s \; \forall \; s \in \{1, 2, ..., M\}$), see Figure 6.2a. Let $C_s = \hat{K}_X^{-1} Y_s$ be the mapping parameterization. Hence, we can rewrite the posterior Eq 6.1 as

$$y^*|x^*, C_s^* \sim \mathcal{N}(\psi_*^\top C_s^*, k_x(x^*, x^*) - \psi_*^\top \hat{K}_X^{-1} \psi_*) \tag{6.2}$$

Because $\hat{K}_X^{-1}$ is square $N \times N$ matrix, C_s has the same dimensionality as the instance sequence Y_s. We can think about C_s as the projection of the points in the instance manifold $\mathcal{M}_s$ into the configuration space of the points in the content manifold. In the same time C_s encodes the style variability between the instance manifolds. Therefore, we can use C_s for parameterization of the mapping $\mathcal{U} \rightarrow \mathcal{M}_s$. However, the dimensionality of this parameterization is huge, which makes it difficult or impossible for continuous inference in the parameterization space. Having the concise (informative and low-dimensional) representation for the style space is essential for robust inference and out of sample generalization.

To find this concise representation, we embed the parameterizations into low-dimensional space. We want to embed $\{C_1, C_2, ..., C_M\}$ into low-dimensional

representation $\{t_1, t_2, ..., t_M\}$, where $t_i \in \mathbb{R}^{d_t}$ and $d_t \ll ND$. We use GPLVM for doing that by building the mapping $\mathcal{C} \sim \mathcal{GP}(\mathbf{0}, K_T)$, where $N \times M \times D$ tensor $\mathcal{C} = [C_1, C_2, ..., C_M]$ be the stacking of parameterizations for all instance manifolds. The posterior distribution can be written as

$$C^*|t^*, \mathcal{C} \sim \mathcal{N}(\mathcal{C} \times_2 \hat{K}_T^{-1}\phi_*, K_T(t^*, t^*) - \phi_*^\top \hat{K}_T^{-1}\phi_*) \tag{6.3}$$

where $\phi(t) = [k_t(., t_1), k_t(., t_2), ..., k_t(., t_M)]^\top$, $\hat{K}_T = (K_T + \lambda_t I)$ and λ_t is the regularization parameter.

Finally, we can get the mapping function $y_s^i = f(x_i, t_s)$, where $x_i \in \mathbb{R}^d$ and $t_s \in \mathbb{R}^\nu$. For doing that, we marginalize out C_s from Eq 6.2 using Eq 6.3 (The superscript $*$ is removed for concise representation),

$$y|x, t = \int \mathcal{N}(\mathbf{y}|\psi^\top C, K(x,x) - \psi^\top \hat{K}_X \psi) \\ \mathcal{N}(C|\mathcal{C} \times_2 \hat{K}_T^{-1}\phi, K_T(t,t) - \phi^\top \hat{K}_T^{-1}\phi)\, dC$$

we get [1]

$$y|x, t \sim \mathcal{N}(\mu_y, \Sigma_y) \tag{6.4}$$

where $\mu_y = \mathcal{C} \times_1 \psi \times_2 \hat{K}_T^{-1}\phi$. Because $\mathcal{C} = \mathcal{Y} \times_1 \hat{K}_X^{-1}$, we can write

$$\mu_y = \mathcal{Y} \times_1 \hat{K}_X^{-1}\psi \times_2 \hat{K}_T^{-1}\phi \tag{6.5}$$

$\mathcal{Y} = [Y_1, Y_2, ..., Y_M]$ is the stacking of all observations in a tensor form of size $N \times M \times D$. $\hat{K}_X^{-1}$ is $N \times N$ matrix and $\hat{K}_T^{-1}$ is $M \times M$ matrix. The covariance is

$$\Sigma_y = K(x,x) - \psi^\top \hat{K}_X^{-1}\psi + \psi^\top [K_T(t,t) - \phi^\top \hat{K}_T^{-1}\phi]\psi \tag{6.6}$$

---

[1] Using the Gaussian identity $\int \mathcal{N}(x|a + Fy, A)\, \mathcal{N}(y|b, B)\, dy = \mathcal{N}(x|a + Fb, A + FBF^\top)$ [Rasmussen and Williams, 2006]

Extension to multiple style factors

Some applications require modeling for more than one style latent factor. We discuss here the extension of Eq 6.4 into two style factors. From Eq 6.3, we can set $\mathbf{g}^* = \mathcal{C} \times_2 \hat{K}_T^{-1}$. Then let $\mathbf{g}^* \sim \mathcal{GP}(\mathbf{0}, K_U)$, we get $y|x,t,u \sim \mathcal{N}(\bar{y}, \Sigma_y)$ where $\bar{y} = \mathcal{Y} \times_1 K_X^{-1}\psi_x \times_2 K_T^{-1}\psi_t \times_3 K_U^{-1}\psi_u$ and $\Sigma_y = K_x(x,x) - \psi_x^\top K_x^{-1}\psi_x + \psi_x^\top[K_t(t,t) - \psi_t^\top K_T^{-1}\psi_t + \psi_t^\top[K_u(u,u) - \psi_u^\top K_U^{-1}\psi_u]\psi_t]\psi_x$

6.5 Style and Content Separation

To find the embedding of a new observation point y, we need to find the content embedding x and the style embedding t that maximize the likelihood Eq 6.4. For the set of observations $\mathbf{Y} = \{y_1^1, y_2^1, ..., y_N^1, y_1^2, y_2^2, ..., y_N^2, ..., y_1^M, y_2^M, ..., y_N^M\}$, we need to find the content embedding ($X = [x_1, x_2, ..., x_N]^\top$) and style embedding ($T = [t_1, t_2, ..., t_M]^\top$), to maximize the total log likelihood of all observed points given the current embedding, so we need to solve

$$(X,T) = \arg\max_{(X,T)} \sum_{j=1}^{M}\sum_{i=1}^{N} \log P(y_i^j | x_i, t_j) \tag{6.7}$$

The Log Likelihood (LL) of a single observation, y, for a potential embeddings x and t, can be written as:

$$LL = -\frac{D}{2}log(2\pi) - \frac{log|\Sigma_y|}{2} - \left(\frac{1}{2}(y-\mu_y)\Sigma_y^{-1}(y-\mu_y)^\top\right) \tag{6.8}$$

where μ_y and Σ_y are defined in Eq 6.5 and Eq 6.6 respectively.

For learning the model and finding this embedding, we propose iterative algorithm that results in unified content embedding (X) and style embedding (T). This is in contrast to multi-factor GPLVM (MGP) [Wang et al., 2007], which optimizes over the Cartesian product space $P = X \otimes T$. Then they proposed

the mapping $Y \sim \mathcal{GP}(\mathbf{0}, K_P)$, where $K_P = K_X \odot K_T$, is Hadamard product of the two kernels. This method does not help to find the common topology. As described in Section 6.2.

Two-way Iterative Algorithm for Learning

We propose approximation algorithm that builds on top of GPLVM to infer the style and content embedding. In Eq 6.5 and Eq 6.6, if we fix style embedding $T = [t_1, ..., t_M]^\top$, then for arbitrary content embedding x we get

$$\mu(x, T) = \mathcal{T} \times_1 \hat{K}_X^{-1} \psi(x)$$
$$\Sigma(x, T) \approx K(x, x) - \psi^\top \hat{K}_X^{-1} \psi$$

where

$$\mathcal{T} = \mathcal{Y} \times_2 \hat{K}_T^{-1} \Phi \tag{6.9}$$

$\mu(x, T)$ and $\Sigma(x, T)$ are the mean and covariance for the posterior distribution of the mapping: $X \to \mathcal{T}$,

$$\mathcal{T} \sim \mathcal{GP}(0, K_X) \tag{6.10}$$

we call this mapping *content-GP*.

Similarly, if we fix the content embedding $X = [x_1, ..., x_N]^\top$, then for arbitrary style embedding we get *style-GP* ($T \to \mathcal{X}$)

$$\mathcal{X} \sim \mathcal{GP}(0, K_T). \tag{6.11}$$

where

$$\mathcal{X} = \mathcal{Y} \times_1 \hat{K}_X^{-1} \Psi \tag{6.12}$$

Inspired by idea of Coordinate Gradient Descent (CGD), from arbitrary initial embedding in the style space, we can compute $\mathcal{T}$ in Eq 6.9. Then we can

use GPLVM to learn the content-embedding from Eq 6.10. Using the optimized content embedding, we can compute $\mathcal{X}$ in Eq 6.12 then learn the style embedding using Eq 6.11. We iterate over this until convergence.

Algorithm 4 summarizes the steps. It uses GPLVM as black-box in iterative way to infer the correct embeddings. In each iteration, the improvement in the content embedding contributes to improve the style embedding, by adjusting $\mathcal{X}$. Similarly, the enhancement in style embedding is propagated to content embedding by adjusting $\mathcal{T}$.

Algorithm 4 GPDS Algorithm for learning the model

Preprocessing: Centralize the point in each sequence
Initialization
Let X_0 be any arbitrary initialization for the content embedding.
Let T_0 be any arbitrary initialization for the style embedding.
repeat
  Compute $\mathcal{T}_i \leftarrow \mathcal{Y} \times_2 K_t(T_{i-1})^{-1}\Phi_{i-1}$
  Let $\mathcal{T}_i \sim \mathcal{GP}(0, K_{X_i})$ (*Content-GP*)
  Use GPLVM to find the content embedding X_i
  Compute $\mathcal{X}_i \leftarrow \mathcal{Y} \times_1 K_x(X_i)^{-1}\Psi_i$
  Let $\mathcal{X}_i \sim \mathcal{GP}(0, K_{T_i})$ (*Style-GP*)
  Use GPLVM to find the style embedding T_i
until Convergence
Retrun The dynamics content X, and the style embedding T

6.6 Inference

Point embedding

To find the style (t^ν) and content (x^ν) embedding of a point y^ν in the observation space. We need to maximize the joint loglikelihood Eq 6.8. $(x^\nu, t^\nu) = \arg\max_{(x,t)} \log P(y^\nu | x, t)$. We used MCMC sampling to optimize this objective.

Sequence embedding

having the unified content trajectory helps to find embedding for image sequence as follows. Two main propoerties for image sequences, first sequence of images lies on single dimensional manifold. Second, all images is required to have the same style embedding. Therefore, to find the embedding of new sequence $Y^\nu = [y_1^\nu, y_\nu^1, ..., y_{n^\nu}^\nu]$, we need to find unique style embedding t^ν, and n^ν content embeddings parameterized by $\Lambda^\nu = [\lambda_1^\nu, \lambda_2^\nu, ..., \lambda_{n^\nu}^\nu]$ that maximizes the sum of log-likelihood Eq 6.8 of all points

$$(X^\nu, t^\nu) = \arg\max_{(\Lambda,t)} \sum_{i=1}^{n^\nu} \log P(y_i^\nu | x(\lambda_i), t) \tag{6.13}$$

where $\lambda_i \in R$ is index variable that search the latent content trajectory X (learned by Algorithm 4) for the best match, and $X^\nu = X(\Lambda^\nu)$. Therefore, our approach increases the efficiency of the inference and makes it insensitive to the dimensionality of the content space. Assuming, we have learned the optimum embedding X, our proposed approach is robust for finding the solution.

For optimizing Eq 6.13, we search the Euclidean space of $(\Lambda, T) \in \mathbb{R}^{d_t+n^\nu}$. We can use sampling based search method such as MCMC sampling such as Metropolis Hastings or Stochastic Annealing. We can also use gradient based method such as Stochastic Conjugate Gradient (SCG). For SCG, we need the derivative of Eq 6.8. For completeness, the derivative is provided in the supplementary materials of this work, in addition to more elaboration about the inference.

6.7 Experimental Results

In this section we show the results of several experiments to show the applicability and effectiveness of the proposed framework. We first introduce the

datasets, then provide the results for style/content embedding in several application, then the inference quantitative results. Despite the great effort behind building the GPLVM toolbox, it suffers from remarkable performance issues. Therefore, to increase the efficiency of our model, we modified the implementation of number of the core files in GPLVM just to increase the learning speed. We also use spherical covariance (isotropic GPs). Even though, this limit the flexibility of the learned models, this significantly reduce the number of parameters and helps to improve the performance. We discovered empirically that very few iterations ($\leq$ *four*) in Algorithm 4 is sufficient to learn the model.

6.7.1 Datasets

Synthetic dataset

Figure 6.1a shows examples of the synthetic dataset, see Section 6.1. The curves are projected orthogonally to ten-dimensional Euclidean space. To ensure point correspondence among all instance manifolds, we restrict the centers to be located at the origin, and all of them are projected to the same 3D subspace in the 10-Dim Euclidean space. We use this controlled dataset to facilitate visualizing the content embedding. We use 3D spaces for both the content and style spaces. We use Radial Basis Function (RBF) kernel for the content-GP (Eq 6.10), and Linear Kernel for the style-GP(Eq 6.11), see Algorithm 4.

3D Objects dataset

[Savarese and Fei-Fei, 2007] contains objects from 10 different categories: car, stapler, iron, shoe, monitor, computer mouse, head, bicycle, toaster and cellphone. Each object is imaged from 24 poses on a viewing sphere (8 azimuth

angles $\times$ 3 zenith angles), and from 3 scales. We used the entire (all classes) 3DObjects dataset to evaluate the performance of the proposed framework on both object categorization and viewpoint estimation. We excluding heads and monitors classes and excluded the farthest scale. This dataset is not designated for motion or dynamics. We use this dataset to visualize the embedding of high dimensional observation space, since HoG features is extracted from this dataset live in about 6K-dimensional space.

Human Motion Capture (MOCAP) dataset [Leordeanu and Hebert, 2005]

It has sets of human locomotion such as walk, jog, jump and run. Each motion is performed by multiple subjects. Each frame in every motion sequence has information about the joint locations and motion. This dataset has different kind of representations. In this dissertation, we use *ASF/ACM* format. Each subject has *ASF* file for the skeleton information and *ACM* file for join motions. We use walk and jog data of subject-35, this split is commonly used in the literature. The sequences in this dataset have different number of points. For each split, we use "interp1" MATLAB function with "PCHIP" mode for interpolating all sequences to unify the sequence length to the max possible length of 71 frames per sequence. Other than that, we don't do any preprocessing or any time warping.

AvLetters dataset [Matthews and Cootes, 2002]

This visual speech dataset has ten subjects repeating every English letter ($A \cdots Z$) three times, with a total of 780 video sequences. For every frame, we extracted **LBP** [Ojala, 2002] features with vector of size $\mathbb{R}^{472}$, and every sequence has exactly 40 frames.

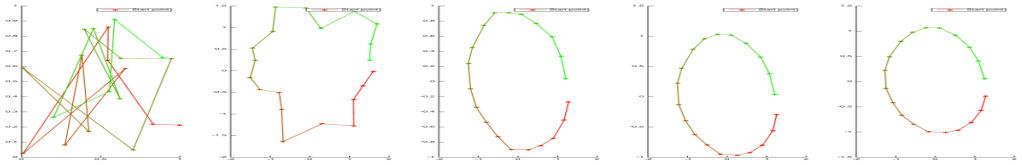
HumanEva-I dataset

[Sigal and Black, 2006], consists of four subjects doing five predefined actions: Walking, Jogging, Throw-catch, Gestures and Boxing. Video data and ground truth motion of the body are captured using a marker-based motion capture system and synchronized in software. The motions is captured with five cameras three color and two black and white. In our experiments, we used HoG features extracted from the color images as used in [Bo and Sminchisescu, 2010].

6.7.2 Embedding and visualization

Embedding the synthetic dataset

Figure 6.1a shows the content and the styles embedding for the 50 instance manifolds of the the synthetic data. Clearly, the content reflects the actual topology of the curves. While, in the style space, we can see the apparent clustering between the five different shapes and styles. Figure 6.3 shows the evolution of the content and style embedding, generated by Algorithm 4. In Figure 6.3a, we can see that the content embedding progressively captures the latent topology of the instance manifolds, and Fig 6.3b shows how the confusion matrix is adapting the real similarity between different manifolds. In Figure 6.3b, we can see strong contrast in the upper-right corner of the confusion matrix, which indicates well clustering of the points that belong to the fifth class (represented by black up-side down triangles in Figure 6.1c). This is because class 5 has the most complex and deformed set of manifolds.



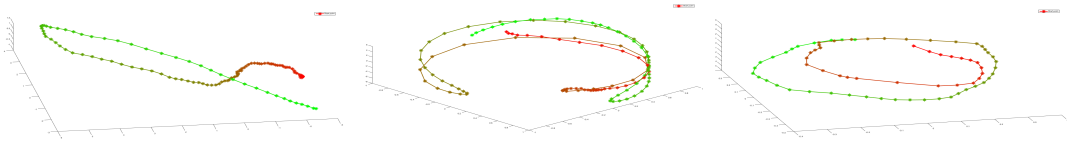
(a) Content-Embedding Evolution: starting from arbitrary random embedding, the actual underlaying topology is infered.

(b) Style-confusion Evolution: confusion matrices for the sequcence of style embeddings.

Figure 6.3: The embedding evolution of the of the entire curves in the synthetic dataset. In four iterations, Algorithm 4 results the correct underlaying topology (content) and representing confusion matrix for style embedding.

Embedding and visualizing the motion capture (MOCAP) dataset

Figure 6.4 shows the embeddings of the MOCAP data in the dynamics/content space. First, we used the same configuration, as been used with the synthetic data: 3D dynamics space with RBF-kernel for content-GP, and 3D motion style with Linear-kernel for style-GP. Figure 6.4a shows the resulting embeddings. Because, locomotion is periodic motion, we use the approach in [Urtasun et al., 2008] to enforce this prior knowledge and constraint the content topology to be periodic. More specifically, we build compound kernel of two kernels for the content-GP: RBF-periodic over the first dimension and linear kernel over the second dimension. This configuration shows how far the proposed framework is customizable to different situations. The resulting embeddings is shown in Figure 6.4b. The third setup is 3D Linear kernel for content-GP, and 3D Linear kernel for style-GP. Despite the simple configuration used in this setup, the content embedding captures most of the motion latent topology, see Figure 6.4c.

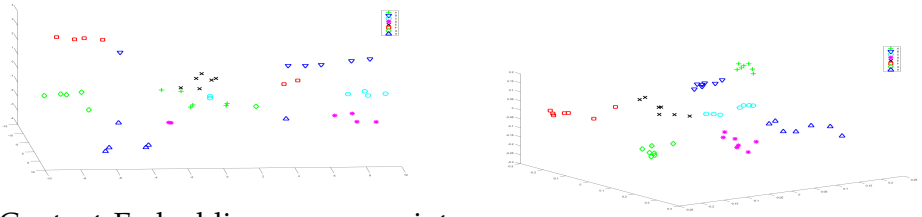


(a) RBF (b) RBFPeriodic (c) Linear

Figure 6.4: MOCAP: Content embedding for the three different kernels for the content-GP: RBF kernel **(a)**, RBF-Periodic + Linear kernel **(b)** and Linear Kernel **(c)**

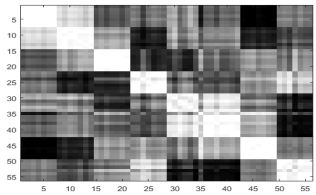
Embedding for Multi-view objects (3DObjects) dataset

Fig 6.5 and Fig 6.6 show embedding of 3D objects dataset using different kernel configurations. In the content space, there is point for the 8 views and each height and scale (labeled by the eight views). For style latent, there point for each instance (labeled by eight category)

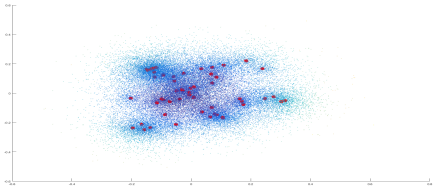


(a) Content Embedding - every point for specific view - hight - scale, labeled by views

(b) Style Embedding - point for each instance, labeled by category

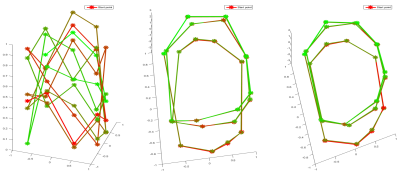


(c) Confusion Matrix of Style Embedding - shows clear clustering

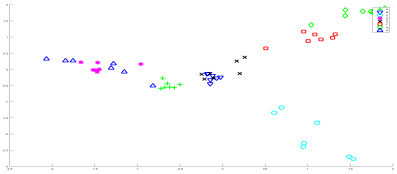


(d) Marginal Covariance for Style Embedding - shows lower covariance close to embedding points

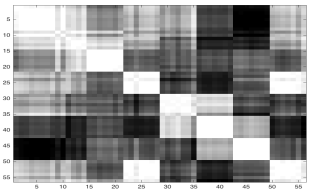
Figure 6.5: 3DObjects Dataset: Compound kernel (2D-Rbf + 1D-Linear) for Content-GP and 3D-Linear kernel for Style-GP.



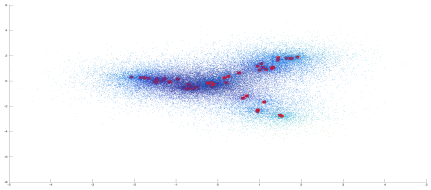
(a) Content Embedding Evolution starting from random - every point for specific view - hight - scale, labeled by views



(b) Style Embedding - point for each instance, labeled by category



(c) Confusion Matrix of Style Embedding - shows clear clustering

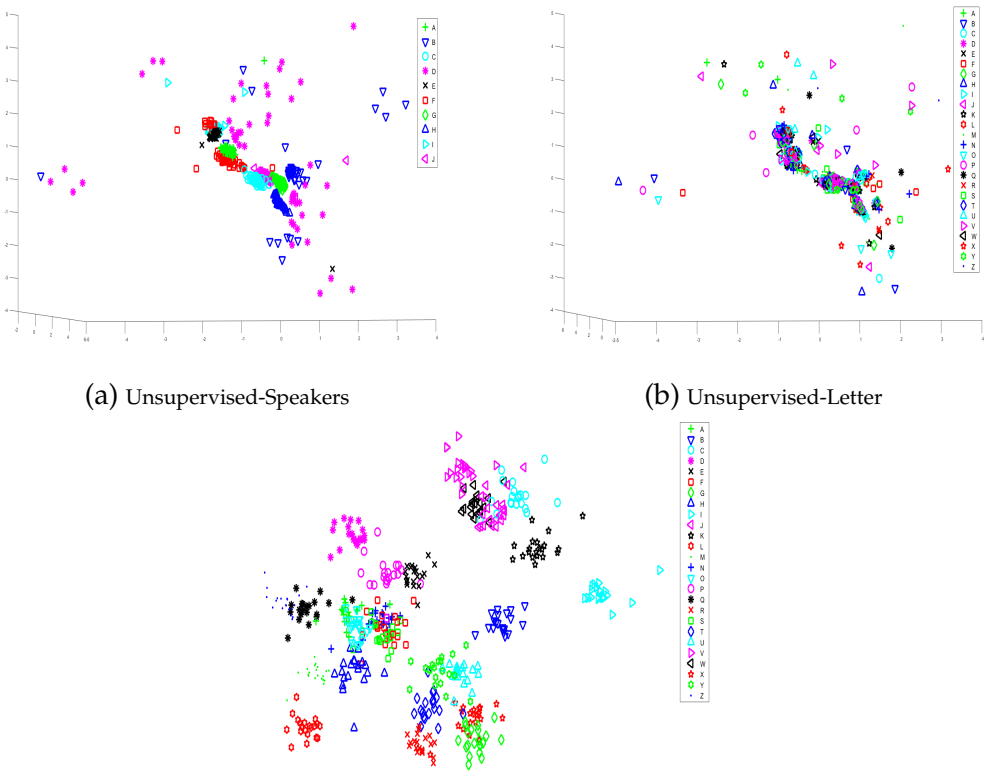


(d) Marginal Covariance for Style Embedding - shows lower covariance close to embedding points

Figure 6.6: 3DObjects Dataset: 1D-RBFPeriodic kernel for Content-GP and 3D-RBF kernel for Style-GP.

Supervised vs unsupervised embedding on AVLetters

The inference in this part is based on the embedding in the style space. To show that, we use 520 sequences in AVLetters database to learn the model. In this model, we used 3D style space, to allow for embedding visualization. Figure 6.7 shows the embedding of all the 520-spoken words in the style space. Figure 6.7a shows the embedding when labeled by the speaker identity. It is clear that the points belong to the same speaker are well clustered. This clustering disappeared when label the same points with the letters Figure 6.7b. This is because the speaker style dominates the letter style, which a known problem in the speech recognition literature. This domination because the speaker visual mouth features is an affecting factor, and the speaker mouth motion is strong biometric in the visual speech datasets. Therefore, there is a need at this point



(a) Unsupervised-Speakers

(b) Unsupervised-Letter

(c) Supervised-Letter

Figure 6.7: AVLetters: Style Embedding of the speech sequences. **(a,b)** are for unsupervised embedding of the training sequences. Speaker's labels used to color the points in **(a)**, and letter's labels used in **(b)**. Supervised embedding using PLS prior with letter labels **(c)**.

to consider supervision when learning the model. However, supervision is beyond the scope of this work, we need to show the advantage of Algorithm 4 being build on top of GPLVM[Lawrence, 2005], so it inherits its powerful functionality. More specifically, in this case, we need to add supervision to the GP-style mapping Eq 6.3 to enforce the data embedding to be maximally correlated with labels. Figure 6.7c shows the embedding in the style space after encoding the supervision by deploying prior. This prior enforces the correlation with the letter labels, [Urtasun and Darrell, 2007].

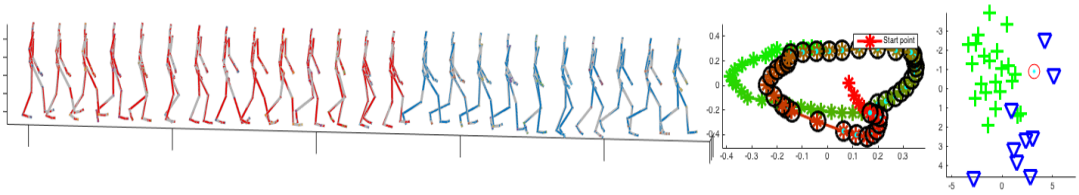


Figure 6.8: MOCAP: predicting the future postures for out of sample motion subsequence. The given subsequence colored in red in the left column. The blue skeletons in the *Left* is for the predicted cycle. For the given subsequence, the content embedding is presented in the *Middle* column, and the style embedding in the *Right*.

6.7.3 Motion style recognition and synthesizing on MOCAP

To show the generative ability of the learned model, we conduct two experiments on CMU MOCAP dataset. First, we learned a model based on 31 motion sequences for Subject-35, 9 of them jogging and 22 walking. Figure 6.8 shows the content embedding (middle) and the style embedding (right) for the given sub-sequence.

For part of walk-subsequence, Figure 6.8 visualize the given sub-sequence (red skeletons) and the generated sub-sequence (blue skeletons). We can see that the generated sub-sequence captures the motion style (Walking - in this case). To infer the content embedding for this sequence, we optimize Eq 6.13. However, instead of optimizing for every value of λ_i, we optimize over the start point and the pace, see Section 6.6. Therefore, the optimization is done in Euclidean space of dimension $\mathbb{R}^{d_t+2}$. *In the supplemental materials, more results will be included for Mocap dataset.*

Second, to compare with MGP, we replicate the experiment described in [Wang et al., 2007]. Six sequences are taken (three walking, two jogging and single stride motion). Each of them down-sampled by a factor of 4, resulting in 314 frames in total. For each frame, 89D feature vector has been extracted. The sequence length has been unified to 78 frames using the same interpolation

techniques described before. We learned the model with basic configuration: 3D Linear kernel for content-GP and 3D RBF-kernel for style-GP.

The Root Mean Square (RMS) error of the predicted sub-sequence is measured in two modes. Long mode, where the extrapolated sub-sequence is 40-frames length for walking and stride case and 20-frames are extrapolated for the running sequences. In this mode, 9 sequences has been used the result of this mode is listed in Table 6.1. The results are compared to the results of three different models: GPDM[Wang et al., 2005], Multi-factor (MF) GPDM and Multi-factor Cyclic Dynamics Models (CDM) [Wang et al., 2007]. The Multi-factor models use stylistic version which uses a style space for each variational parameter (the gait style and the subject identity) besides the content latent which is called the motion state in this work. Our framework outperform all other model except for the stylistic CDM in some cases.

Table 6.1: Comparison with MGP[Wang et al., 2007]: RMS errors for long prediction. Sequence name is the file name in CMU MOCAP dataset

Model		GPDM	MF-GPDM	MF-CDM	Ours
Sequence	Motion Style				
07-02	walk	1.56	0.91	0.38	0.66
08-04	walk	1.18	0.48	0.47	0.70
08-05	stride	1.91	0.56	1.77	0.57
08-11	stride	2.42	1.06	0.80	0.71
07-04	walk	1.10	1.10	0.72	0.65
07-12	walk	1.45	1.06	0.57	0.92
37-01	walk	1.04	0.75	0.35	0.56
16-35	jog	1.41	0.53	0.39	0.70
09-07	jog	1.34	0.49	0.57	0.74
Average		1.49	0.77	0.67	0.69

The second mode is the short mode. In which, 20-frames are extrapolated for walk sequences given only 10-frames. Table 6.2 shows the results of this mode. Our framework significantly outperforms the other models.

Table 6.2: Comparison with MGP[Wang et al., 2007]: RMS errors for short prediction.

Model		B-GPDM	Mf B-GPDM	Mf CDM	Ours
Sequence	Motion Style				
07-04	Walk	1.21	0.92	1.12	0.75
07-12	Walk	1.48	0.88	1.14	0.76
37-01	Walk	1.00	0.70	0.85	0.72

Efficient Algorithm

In addition to outperforming the results (in most cases) the proposed framework is more flexible and significantly more efficient. As a matter of comparison, for learning the described model based only on six sequences, our model takes about 7.6 **seconds** to learn the embeddings using three rounds, in every round, two GPLVM models (content-GP and style-GP) are optimized[2] , see Algorithm 4. While, MGP (using their published code) is setup to 10 rounds. Using non-gradient optimization mode, every single round takes on the average 4599.3 **seconds**, with total of about ten hours to process this few number of sequences. We understand that the comparison here is not fair, because we use two different implementations. However, it gives impression about how efficient is our approach. However, this extreme performance of the MGP framework does not allow to compare in different applications. Other quantitative results will appear in the supplemental materials.

6.7.4 Style-based inference for speech recognition

One potential application of this framework is the recognition of a given instance manifold. We performed speaker identification on AVLetters dataset. The problem is for a given speech sequence, we need to recognize the speaker

[2] We use the published GPLVM toolbox with the aforementioned performance issues.

Table 6.3: AVLetters: Speaker Recognition.

Speaker#	1	2	3	4	5	
Rate	100%	100%	96.15%	92.31%	100%	
Speaker#	6	7	8	9	10	Overall
Rate	100%	100%	100%	100%	92.31%	98.08%

identity. This problem is known in the literature by speaker identification. For this experiment, we used the model learned in Section 6.7.2, we used the remaining 260 sequences for testing. The speaker identification gives 98.08% recognition accuracy, Table 6.3 shows the accuracy of recognizing each speaker. The results show slight confusion in recognizing of the fourth speaker, since his/her sequences don't cluster well in the style space Figure 6.7a.

6.7.5 Content-based inference for human pose estimation

To show one important application for inference on the content space, we used Human-Eva-I dataset for human pose estimation (HPE). In this work, we use simple technique to show how to use the learned unified content trajectory for inference. For each one of the 15 sequences, we learned the model based on the three cameras of the training part as different styles. Content represents time, style is for cameras and observation space is HoG feature of single camera image. This configuration guarantees perfect alignment between the three sequences (one for each camera). For test, we infer style and content for each camera individually. For inference, we used the algorithm in Section 6.6. To estimate the pose, we use corresponding pose to the closest point in the content trajectory. This split gives us unified content manifold which is the motion trajectory in the pose space, and several styles for different cameras. However, it is not comparable to any of the previously reported results based on

Table 6.4: HumanEva I: Human Pose Estimation. Numbers are RMS in mm. Split description in the text

	Boxing	Gesture	Jogging	Catch-Throw	Walking
Subject-1	102	25	156	118	132
Subject-2	124	159	169	168	191
Subject-3	193	46	179	–	210

this dataset. Table 6.4 shows the results for the 15 sequences ordered in subjects and motion styles. More results on this dataset will be in supplemental materials.

6.8 Conclusion and Future Work

In this work, we build a novel multi-factor probabilistic generative model, based on Gaussian processes, for modeling the dynamical behavior of multi instance datasets.We derived closed form for the likelihood of points in the input space given points in the content and style spaces. We show the extension to model more than single style factor. In addition, we proposed iterative algorithm, on top of GPLVM, that embeds the input data into two separate style and content spaces. The framework inherits the flexibility and richness of the GPLVM. We applied the approach on a synthesized dataset, and we being able to find the hidden topology of the manifolds. Moreover, we applied the framework on CMU MOCAP dataset, and we show that our approach helps to capture the actual motion topology in the MOCAP dataset. This work is extend-able in many directions: it is applicable to motion classification, as we showed preliminary results on AVLetters speech recognition dataset. We also used simple technique to show the applicability of our framework to solve human pose estimation. Because of its efficiency, using this framework can

leverage the use of GPLVM in many computer vision applications. The model has many extension to be used with many motion sequences without having to unify the sequence length.

Chapter 7

Digging Deep into the Layers of CNNs: In Search of How CNNs Achieve View Invariance

7.1 Introduction

Impressive results have been achieved recently with the application of Convolutional Neural Networks (CNNs) in the tasks of object categorizations [Krizhevsky et al., 2012] and detection [Sermanet et al., 2013; Girshick et al., 2013]. Several studies recently investigated different properties of the learned representations at different layers of the network, *e.g.* [Yosinski et al., 2014; Zeiler and Fergus, 2013; Chatfield et al., 2014]. One fundamental question is how CNN models achieve different invariances. It is well understood that consecutive convolution and pooling layers can achieve translation invariant. Training CNN networks with a large dataset of images, with arbitrary viewpoints and arbitrary illumination, while optimizing the categorization loss helps to achieve viewpoint invariant and illumination invariant.

In this work, we focus on studying the viewpoint invariant properties of CNNs. In many applications, it is desired to estimate the pose of the object, for example for robot manipulation and scene understanding. Estimating pose and object categorization are tasks that contradict each other; estimating pose requires a representation capable of capturing the viewpoint variance, while

viewpoint invariance is desired for categorization. Ultimately, the vision system should achieve a representation that can factor out the viewpoint for categorization and preserve viewpoint for pose estimation.

The biological vision system is able to recognize and categorize objects under wide variability in visual stimuli, and at the same time is able to recognize object pose. It is clear that images of the same object under different variability, in particular different views, lie on a low-dimensional manifold in the high-dimensional visual space defined by the retinal array ($\sim$100 million photoreceptors and $\sim$1 million retinal ganglion cells). DiCarlo and Cox [2007] hypothesized that the ability of our brain to recognize objects, invariant to different viewing conditions, such as viewpoint, and at the same time estimate the pose, is fundamentally based on untangling the visual manifold encoded in neural population in the early vision areas (retinal ganglion cells, LGN, V1). They suggested that this is achieved through a series of successive transformation (re-representation) along the ventral stream (V1,V2, V4, to IT) that leads to an untangled population at IT. Despite this, it is unknown how the ventral stream achieves this untangling. They argued that since IT population supports tasks other than recognition, such as pose estimation, the manifold representation is some how *'flattened'* and *'untangled'* in the IT layer. DiCarlo and Cox's hypothesis is illustrated in Figure 7.1. They stress that the feedforward cascade of neural re-representation is a way to untangle the visual manifold.

Inspired by recent impressive results of CNNs and by DiCarlo and Cox's hypothesis [DiCarlo and Cox, 2007] on manifold untangling, this chapter focuses on studying the view-manifold structure in the feature spaces implied by the different layers of CNNs. There are several questions that this study aims to answer: **1.** Does the learned CNN representations achieve viewpoint

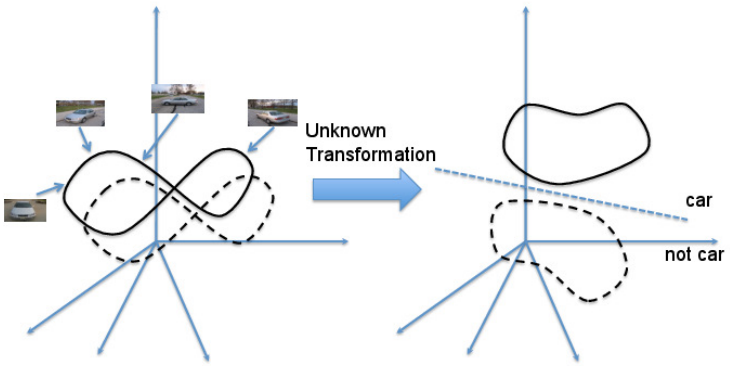


Figure 7.1: Illustration of DiCarlo and Cox model [DiCarlo and Cox, 2007]: Left: tangled manifolds of different objects in early vision areas. Right: untangled (flattened) manifold representation in IT

invariance? If so, how does it achieve viewpoint invariance? Is it by collapsing the view manifolds, or separating them while preserving them? At which layer is the view invariance achieved? **2.** How to experimentally quantify the structure of the viewpoint manifold at each layer of a deep convolutional neural network? **3.** How does fine-tuning of a pre-trained CNN, optimized for categorization, on a multi-view dataset, affect the representation at each layer of the network?

In order to answer these questions, we present a methodology that helps to get an insight about the structure of the viewpoint manifold of different objects as well as the combined object-view manifold in the layers of CNN. We conducted a series of experiments to quantify the ability of different layers of a CNN to either preserve the view-manifold structure of data or achieve a view-invariant representation.

The contributions of the part of the dissertation are as follows:

- We propose a methodology to quantify and get insight into the manifold structures in the learned representation at different layers of CNNs.
- We use this methodology to analyze the viewpoint manifold of pre-trained CNNs.

- We study the effect of transfer learning a pre-trained network with two different objectives (optimizing category loss vs. optimizing pose loss) on the representation.
- We draw important conclusions about the structure of the object-viewpoint manifold and how it coincides with DiCarlo and Cox's hypothesis.

The chapter begins by reviewing closely related works. Section 7.3 defines the problem, experimental setup, and the basic CNN network that our experiments are based upon. Section 7.4 introduces our methodology of analysis. Sections 7.5 and 7.6 describe the findings on the pre-trained network and the fine-tuned networks respectively. The conclusion section summarizes our findings.

7.2 Related Work

LeCun *et al.* has widely used CNNs for various vision tasks [Sermanet et al., 2013; Kavukcuoglu et al., 2010; Jarrett et al., 2009; Ranzato et al., 2007; LeCun et al., 2004]. The success of CNNs can be partially attributed to these efforts, in addition to training techniques that have been adopted. Krizhevsky et al. [2012] used a CNN in the ImageNet Challenge 2012 and achieved state-of-the-art accuracy. Since then, there have been many variations in CNN architectures and learning techniques within different application contexts. In this section we mainly emphasize related works that focused on bringing an understanding of the representation learned at the different layers of CNNs and related architectures.

Yosinski et al. [2014] studied how CNN layers transition from general to specific. An important finding in this study is that learning can be transferred,

and by using fine-tuning, performance is boosted on novel data. Other transfer learning examples include [Razavian et al., 2014; Donahue et al., 2013; Agrawal et al., 2014]. Zeiler and Fergus [2013] investigated the properties of CNN layers for the purpose of capturing object information. This study is built on the premise that there is no coherent understanding of why CNNs work well or how we can improve them. Interesting visualizations were used to explore the functions of layers and the intrinsics of categorization. The study stated that CNN output layers are invariant to translation and scale but not to rotations. The study in [Chatfield et al., 2014] evaluated different deep architectures and compared between them. The effect of the output-layer dimensionality was explored.

7.3 Problem Definition and Experimental Setup

It is expected that multiple views of an object lie on intrinsically low-dimensional manifolds (*v*iew manifold[1]) in the input space. View manifolds of different instances and different objects are spread out in this input space, and therefore form jointly what we call the *o*bject-view manifold. The input space here denotes the $R^{N\times M}$ space induced by an input image of size $N \times M$, which is analogous to the retinal array in the biological system. For the case of a viewing circle(s), the view manifold of each object instance is expected to be a 1-dimensional closed curve in the input space. The recovery of the category and pose of a test image reduces to finding which of the manifolds this image belongs to, and what is the intrinsic coordinate of that image within that manifold. This view of the problem is shared among manifold-based approaches

[1] we use the terms *view manifold* and *viewpoint manifold* interchangeably

such as [Murase and Nayar., 1995; Zhang et al., 2013; Bakry and Elgammal, 2014]

The ability of a vision system to recover the viewpoint is directly related to how the learned representation preserves the view manifold structure. If the transformation applied to the input space yields a representation that results in collapsing the view manifold, the system will no longer be able to discriminate between different views. Since each layer of a deep NN re-represents the input in a new feature space, the question would be how the re-representations deform a manifold that already exists in the input space. A deep NN would satisfy the hypothesis of *'flattening'* and *'untangling'* by DiCarlo and Cox [2007], if the representation in a given layer separates the view manifolds of different instances, without collapsing them, in a way to be able to put a separating hyperplanes between different categories. Typically CNN layers exhibit general-to-specific feature encoding, from Gabor-like features and color blobs at low layers to category-specific features at higher layers [Zeiler and Fergus, 2013]. We can hypothesize that for the purpose of pose estimation, lower layers should hold more useful representations that might preserve the view manifold and be better for pose estimation. But which of these layers would be more useful, and where does the view-manifold collapse to view-invariance.

There are different hypotheses we can make about how the view manifolds of different objects are arranged in the feature space of a given layer. These hypotheses are shown in Figure 7.2. We arrange these hypotheses based on linear separability of the different objects' view manifolds and the preservation of the view manifolds. Case 0 is the non-degenerate case where the visual manifolds preserve the pose information but are tangled and there is no linear separation between them (this might resemble the input space, similar to left

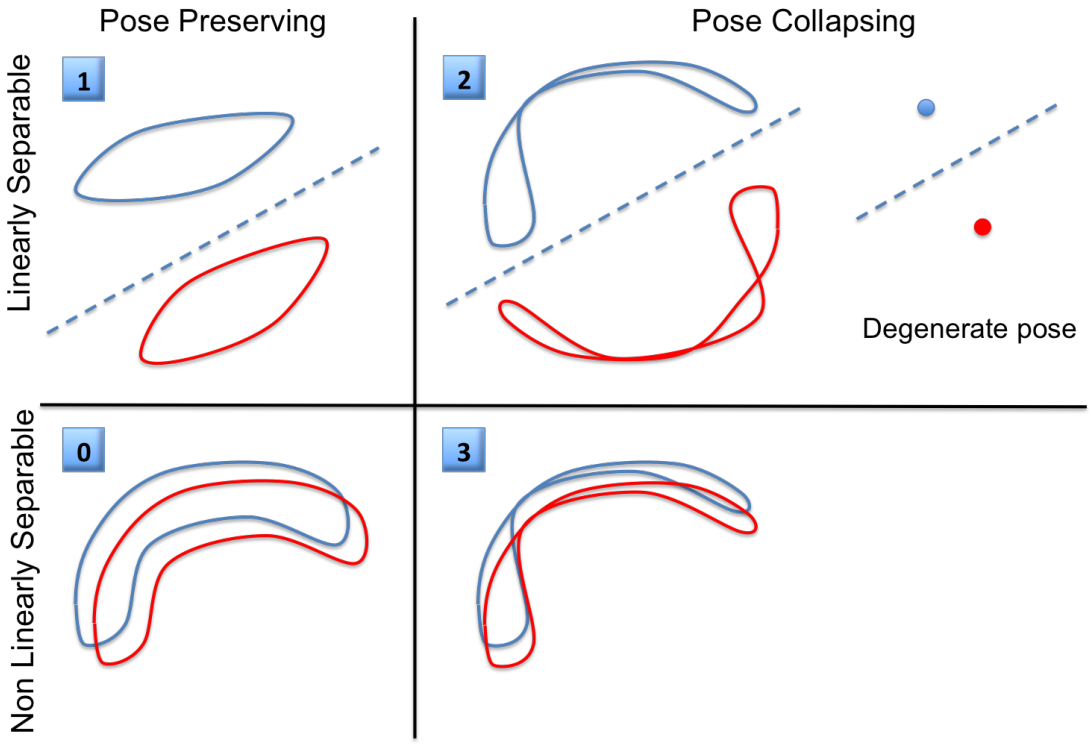


Figure 7.2: Sketches of four hypotheses about possible structures of the view manifolds of two objects in a given feature space.

case in Figure 7.1). Case 1 is the ultimate case where the view manifolds of different objects are preserved by the transformation and are separable (similar to the right case in Figure 7.1). Case 2 is where the transformation in the network leads to separation of the object's view manifold at the expense of collapsing these manifolds to achieve view invariance. Collapsing of the manifolds can be to different degrees, to the point where each object's view manifold can be mapped to a single point. Case 3 is where the transformation results in more tangled manifolds (pose collapsing and non-separable). It is worth to notice that both cases 1 and 2 are view invariant representations. However, it is obvious that case 1 would be preferred since it also facilitates pose recovery. It is not obvious whether optimizing a network with a categorization loss result in case 1 or case 2. Getting an insight about which of these hypotheses are true in a given layer of a CNN is the goal of this study. In Section 7.4 we propose a

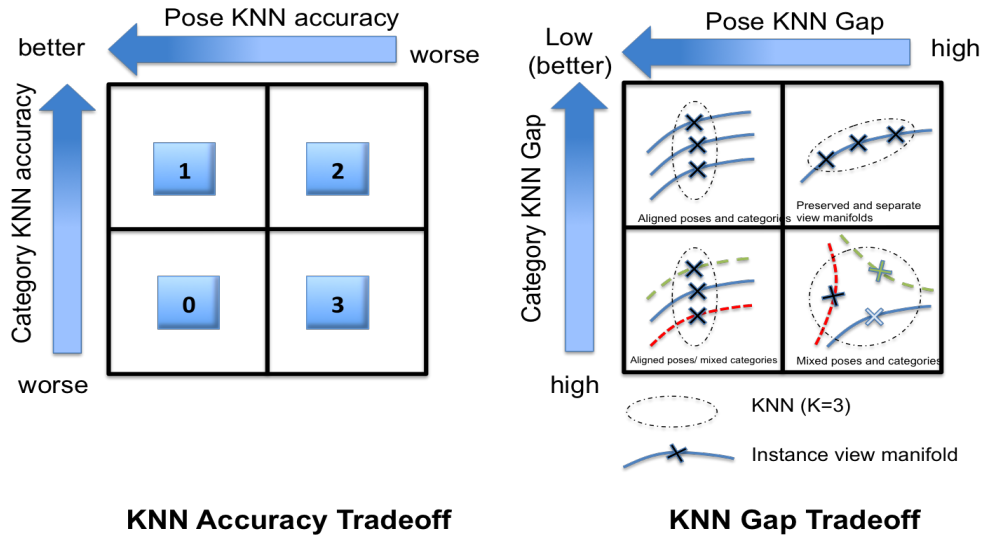


Figure 7.3: KNN Tradeoffs: accuracy tradeoff between category and pose estimation using KNN. This cartoon illustrates the global measurements, see Section 7.4.2 for full details.

methodology to get us to that insight.

7.3.1 Experimental Settings

To get an insight into the representations of the different layers and answer the questions posed in Section 7.1 we experiment on two datasets: I) RGB-D dataset [Lai et al., 2011a], II) Pascal3D+ dataset [Xiang et al., 2014]. We selected the RGB-D dataset since it is the largest available multi-view dataset with the most dense viewpoint sampling. The dataset contains 300 instances of tabletop objects (51 categories). Objects are set on a turntable and captured by an Xbox Kinect sensor (Kinect 2010) at 3 heights (30°, 45° and 60° elevation angles). The dense view sampling along each height is essential for our study to guarantee good sampling of the view manifold. We ignore the depth channel and only used the RGB channels.

Pascal3D+ is very challenging because it consists of images "in the wild", in other words, images of object categories exhibiting high variability, captured in

uncontrolled settings and under many different poses. Pascal3D+ contains 12 categories of rigid objects selected from the PASCAL VOC 2012 dataset [Everingham et al., 2010]. These objects are annotated with 3D pose information (*i.e*, azimuth, elevation and distance to camera). Pascal3D+ also adds 3D annotated images of these 12 categories from the ImageNet dataset [Deng et al., 2009]. The *bottle* category is omitted in state-of-the-art results. This leaves 11 categories to experiment with. There are about 11,500 and 7,000 training images in ImageNet and Pascal3D+ subsets, respectively. For testing, there are about 11,200 and 6,900 testing images for ImageNet and Pascal3D+, respectively. On average there are about 3,000 object instances per category in Pascal3D+, making it a challenging dataset for estimating object pose.

The two datasets provide different aspect of the analysis. While the RGB-D provides dense sampling of each instance's view manifold, Pascal3D+ dataset contains only very sparse sampling. Each instance is typically imaged from a single viewpoint, with multiple instances of the same category sampling the view manifold at arbitrary points. Therefore, in our analysis we use the RGB-D dataset to analyze each instance viewpoint manifold and the combined object-viewpoint manifolds, while the Pascal3D provides analysis of the viewpoint manifold at the category level.

Evaluation Split: For our study, we need to make sure that the objects we are dealing with have non-degenerate view manifolds. We observed that many of the objects in the RGB-D dataset are ill-posed, in the sense that the poses of the object are not distinct. This happens when the objects have no discriminating texture or shape to be able to identify the different poses (*e.g.* a texture-less ball, apple or orange on a turntable). This will cause view manifold degeneracy. Therefore we select 34 out of the 51 categories as objects that possess

pose variations across the viewpoints, and thus are not ill-posed with respect to pose estimation.

We split the data into training, validation and testing. Since in this datasets, most categories have few instances, we left out two random object instances per category, one for validation and one for testing. In the case where a category has less than 5 instances, we form the validation set for that category by randomly sampling from the training set. Besides the instance split, we also left out all the middle height for testing. Therefore, the testing set is composed of unseen instances and unseen heights and this allows us to more accurately evaluate the capability of the CNN architectures in discriminating categories and estimating pose of tabletop objects.

7.3.2 Base Network: Model0

The base network we use is the Convolutional Neural Network described in Krizhevsky et al. [2012] and winner of LSVRC-2012 ImageNet challenge [Russakovsky et al., 2014]. The CNN was composed of 8 layers (including 1000 neuron output layer corresponding to 1000 classes). We call these layers in order: Conv1, Pool1, Conv2, Pool2, Conv3, Conv4, Conv5, Pool5, FC6, FC7, FC8 where Pool indicates Max-Pooling layers, Conv indicates layers performing convolution on the previous layer and FC indicates fully connected layer. The last fully connected layer (FC8) is fed to a 1000-way softmax, which produces a distribution over the category labels of the dataset.

7.4 Methodology

The goal of our methodology is two-folds:

- study the transformation that happens to the viewpoint manifold of a specific object instance at different layers,
- study the structure of the combined object-view manifold at each layer to get an insight about how tangled or untangled the different objects' viewpoint manifolds are.

Both these approaches will get us an insight to which of the hypotheses explained in Section 7.3 is correct at each layer, at least relatively by comparing layers. This section introduces our methodology, which consists of two sets of measurements to address the aforementioned two points. First, we introduce instance-specific measurements that quantify the viewpoint manifold in the different layers to help understand whether the layers preserve the manifold structure. *W*e performed extensive analysis on synthetic manifold data to validate the measures, see Appendix A. Second, we introduce empirical measurements that are designed to draw conclusions about the global object-viewpoint manifold (involving all instances).

7.4.1 Instance-Specific View Manifold Measurements

Let us denote the input data (images taken from a viewing circle and their pose labels) for a specific object instance as $\{(x_i \in \mathbb{R}^D, \theta_i \in [0, 2\pi]), i = 1 \cdots N\}$, where D denotes the dimensionality of the input image to the network, and N is the number of the images, which are equally spaced around the viewing circle. These images form the view manifold of that object in the input space denoted by $\mathcal{M} = \{x_i\}_1^N$. Applying each image to the network will result in a series of nonlinear transformations. Let us denote the transformation from the input to layer l by the function $f_l(x) : \mathbb{R}^D \rightarrow R^{d_l}$ where d_l is the

dimensionality of the feature space of layer l. With an abuse of notation we also denote the transformation that happens to the manifold $\mathcal{M}$ at layer l by $\mathcal{M}^l = f_l(\mathcal{M}) = \{f_l(x_i)\}_1^N$. After centering the data by subtracting the mean, let $\mathbf{A}^l = [\acute{f}_l(x_i) \cdots \acute{f}_l(x_N)]$ be the centered feature matrix at layer l of dimension $d_l \times N$, which corresponds to the centered transformed images of the given object. We call $\mathbf{A}^l$ the sample matrix in layer l.

Since the dimensionality d_l of the feature space of each layer varies, we need to factor out the effect of the dimensionality. Since $N \ll d_l$ the transformed images on all the layers lie on subspaces of dimension N in each of the feature spaces. Therefore, we can change the bases to describe the samples using N dimensional subspace, *i.e*, we define the $N \times N$ matrices $\hat{\mathbf{A}}^l = \mathbf{U}^\mathsf{T}\mathbf{A}$ where $\mathbf{U} \in \mathbb{R}^{d_l \times N}$ are the orthonormal bases spanning the column space of $\mathbf{A}^l$ (which we can get by SVD of $\mathbf{A}^l = \mathbf{USV}^\mathsf{T}$). This projection rotates the samples at each layer without changing the manifold geometric or neighborhood properties. Then the following measures will be applied to the N transformed images, representing the view manifold of each object instance individually. To obtain an overall measures for each layer we will average these measures over all the object instances.

1) Measure of spread - Nuclear Norm:

There are several possible measures of the spread of the data in the sample matrix of each view manifold. We use the nuclear norm (also known as the trace norm [Horn, 2013]) defined as $||\mathbf{A}||_* = Tr(\sqrt{\mathbf{A}^\mathsf{T}\mathbf{A}}) = \sum_{i=1}^N \sigma_i$, *i.e*, it measures the sum of the singular values of $\mathbf{A}$.

2) Subspace dimensionality measure - Effective-p:

Computed by counting the effective dimensionality of the subspace where the view manifold lives. Smaller number means that the view manifold lives in lower dimensional subspace. We define *E*ffective-p as the minimum number of singular values (in decreasing order) that sum up to more that or equal to p% of the nuclear norm, i.e, $\text{Effective} - p = \sup\{n : \sum_{i=1}^{n} \sigma_i / \sum_{i=1}^{N} \sigma_i \leq p/100\}$.

3) Alignment Measure - KTA:

Ideally the view manifold resulting of the view sitting of the studied datasets is a single-dimensional closed curve in the feature space, which can be thought as a deformed circle [**?**]. This manifold can be degenerate in the ultimate case to a single point in case of a texture-less object. The goal of this measurement is to quantify how the transformed manifold locally preserves the original manifold structure. To this end we compare the kernel matrix of the transformed manifold at layer l, denote by $\mathbf{K}_n^l$, with the kernel matrix of the an embedding of the ideal view manifold on unit circle, denote by $\mathbf{K}_n^\circ$, where n indicates the local neighborhood size used in constructing the kernel matrix. We construct the neighborhood based on pose labels.

Given these two kernel matrices we can define several convergence measures. We use *K*ernel Target Alignment (KTA) which has been used in the literature for kernel learning [N et al., 2001]. It finds a scale invariant dependency between two normalized kernel matrices[2]. Therefore, we define the alignment of the transformed view manifold $\mathcal{M}^l$ at layer l with the ideal manifold as

[2] We also experimented with HSIC [Gretton et al., 2005], however HSIC is not scale invariant and not designed to compare data in different feature spaces. Therefore, HSIC did not give any discriminative signal

$KTA_n(\mathcal{M}^l) = < \mathbf{K}_n^l, \mathbf{K}_n^\circ >_F / (||\mathbf{K}_n^l||_F ||\mathbf{K}_n^\circ||_F)$.

4) KPLS-regression measures:

Kernel Partial Least Squares (KPLS) [Rosipal and Trejo, 2002] is a supervised regression method. KPLS iteratively extracts the set of principal components of the input kernel that are most correlated with the output. For more information about KPLS, see Section 2.2. We use KPLS to learn mapping $\mathbf{K}_n^l \rightarrow \mathbf{K}_n^\circ$ from the transformed view manifold kernel (input kernel) to the unit circle kernel (output kernel). We enforce this mapping to use maximum of $d \ll N$ principal components (we used $d = 5$). Then we define *K*PLS-Regression Error, which uses the Normalized Cross Correlation to quantify the mapping correctness. More details in Section 7.7.

5) TPS-linearity measure:

In this measure we learn a regularized Thin Plate Spline (TPS) non-linear mapping [Duchon, 1977] between the unit circle manifold and each $\mathcal{M}^l$. The reason for using TPS in particular is that the mapping has two parts: affine (linear polynomial) and nonlinear part. Analysis of the two parts will tell us if the mapping is mostly linear or nonlinear. We use the *r*eciprocal-condition number (rcond) of the sub coefficient matrices corresponding to the affine and the nonlinear part as a measure of the linearity of the transformation. More details in Section 7.7.

7.4.2 Global Object-Viewpoint Manifold Measures

To achieve an insight about the global arrangement of the different objects' view-manifolds in a given feature (layer) space, we use the following three

empirical measurements:

6) Local Neighborhood Analysis:

To evaluate the local manifold structure we also evaluate the performance of nearest neighbor classifiers for both category and pose estimation, with varying size of the neighborhood. This directly tell us whether the neighbors of a given point are from the same category and/or of similar poses. KNN for categorization cannot tell us about the linear separability of classes. However evaluating the pose estimation in neighborhood of a datapoint gives us an insight about how the view manifolds are preserved, and even whether the view manifolds of different instances are aligned. To achieve this insight we use two different measurements: **KNN-Accuracy**: the accuracy of KNN classifiers for category and pose estimation. **KNN-Gap**: the drop in performance of each KNN classifier as the neighborhood size increases. In our experiments we increase K from 1 to 9. Positive gap indicates a drop (expected) and negative gap indicates improvement in performance.

The interaction between these two measures and how they tell us about the manifold structure is illustrated in Figure 7.3. The contrast between the accuracy of the KNN classifiers for pose and category directly implies which of the hypotheses in Figure 7.2 is likely. The analysis of KNN-Gap (assuming good 1-NN accuracy) gives further valuable information. As the KNN-gap reaches zero in both category and pose KNN classifiers, this implies that neighborhoods are from the same category and has the same pose, which indicates that the representation aligns the view manifolds of different instances of the same category. If the view manifolds of such instances are preserved and separated in the space, and the neighbors of a given point are from the same instance, this

would imply small gap in the category KNN classifier and bigger gap in pose KNN classifier. Low gap in pose KNN vs high gap in category CNN implies the representation aligns view manifolds of instances of different categories. A high gap in both obviously implies the representation is tangling the manifolds such that a small neighborhood contains points from different categories and different poses. Notice that this implications are only valid when the 1-NN accuracy is high.

7) L-SVM:

For a test image x transformed to the l-th layer's feature space, $f_l(x)$, we compute the performance of a linear SVM classifier trained for categorization. Better performance of such a classifier directly implies more linear separability between different view manifolds of different categories.

8) Kernel Pose Regression:

To evaluate whether the pose information is preserved in a local neighborhood of a point in a given feature space we evaluate the performance of kernel ridge regression for the task of pose estimation. Better performance implies better pose-preserving transformation, while poor performance indicates pose-collapsing transformation. The combination of L-SVM and kernel regression should be an indication to which of the hypotheses in Figure 7.2 is likely to be true.

7.5 Analysis of the Pre-trained Network

7.5.1 Instance View Manifold Analysis

Figure 7.4 shows the application of the instance-specific view manifold measurements on the images of the RGBD dataset when applied to a pre-trained network (Model0 - no fine-tuning). This gives us an insight on the transformation that happens to the view manifold of each object instance at each layer of the network. Figure 7.4a shows that the nuclear norm of the transformed view manifolds in Model0 is almost monotonically decreasing as we go higher in the network, which indicates that the view manifolds is more spread in the lower layers. In fact at the output layer of Model0 the nuclear norm becomes too small, which indicates that the view manifold is collapsing to reach view invariant representation at this layer. Figure 7.4b ($p = 90\%$) shows that subspace dimension varies within a small range in the lower layers and it reduces dramatically in fully connected layers, which indicates that the network tries to achieve view invariance. The minimum is achieved at FC8 (even without fine tuning). Figure 7.4c shows the KTA applied to Model0, where we can notice that the alignment is almost similar across the lower layers, with Pool5 having the maximum alignment, and then starts to drop at the very high layers. which indicates that after Pool5, the FC layers try to achieve view invariant. Figure 7.4d shows that KPLS regression error on Model0 dramatically reduces from FC8 down to Pool5, where Pool5 has the least error. In general the lower layers have less error. This indicates that the lower layers preserve higher correlation with the ideal manifold structure. Figure 7.4e shows that the mapping is highly linear, which is expected because of the high dimensionality of the feature spaces. From Figure 7.4e we can clearly notice that the lower layers has

more better-conditioned linear mapping. More details in Section 7.7.

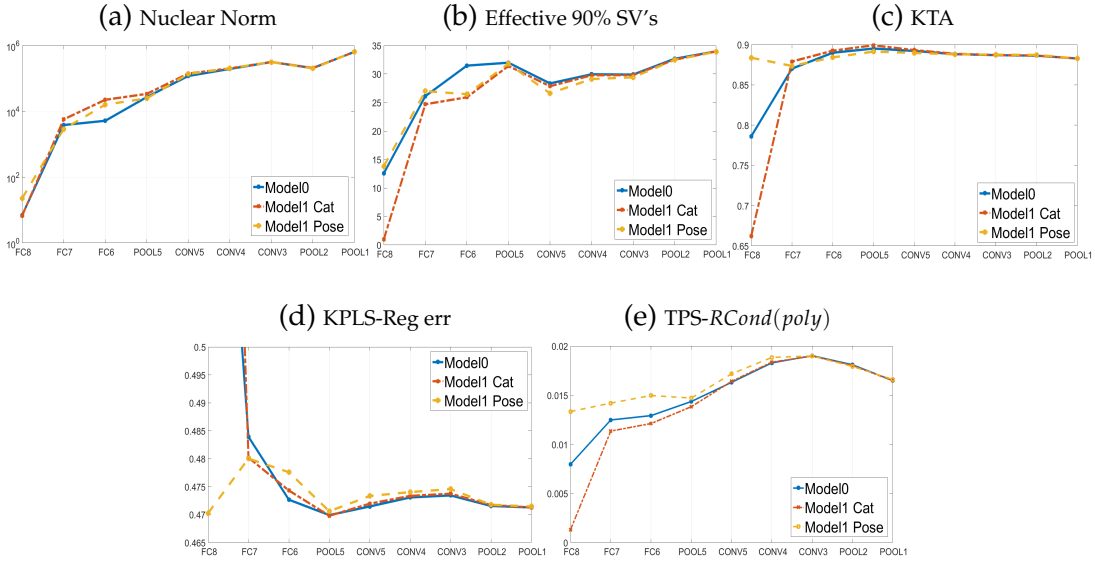


Figure 7.4: RGB-D: Local Measurement analysis for the view-manifold. Every figure shows single measurement for three models (Model0, Model1Cat and Model1Pose) at different layers.

From these measurements we can conclude: (1) The lower layers preserve the view manifolds. The manifolds start to collapse in the FC layers to achieve view invariance. Preserving the view manifold at the lower layers is intuitive because of the nature of the convolutional layers. (2) The manifold at Pool5 achieves the best alignment with the pose labels. This is a less intuitive result; why does the representation after successive convolutions and pooling improves the view manifold alignment? even without seeing any dense view manifold in training, and even without any pose labels being involved in the loss. The hypotheses we have to justify that Pool5 has better alignment than the lower layers is that Pool5 has better translation invariant properties, which results in improvement of the view manifold alignment.

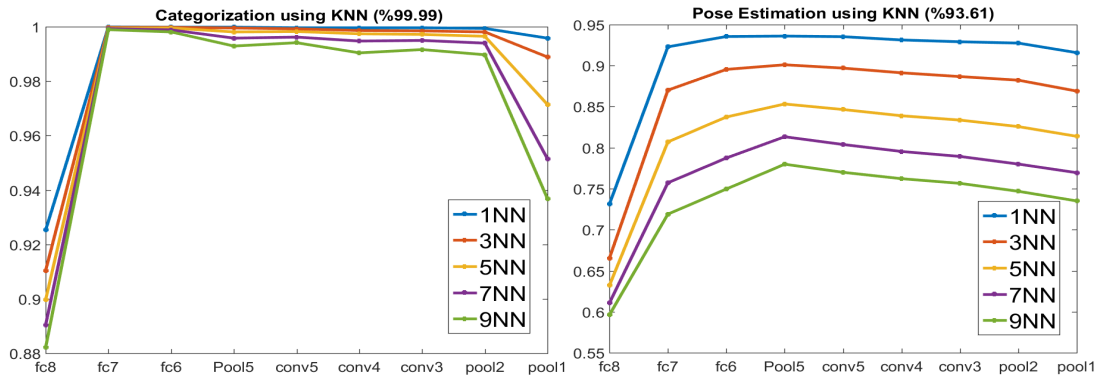


Figure 7.5: RGB-D: KNN for categorization and pose estimation over the layers of pre-trained model (Model0). For $K = \{1, 3, 5, 7, 9\}$

.

7.5.2 Global Object-View Manifold Analysis

To study view-manifold in the network layers, Figure 7.5 shows the KNN accuracy for pose and category within training split, no test is used in this experiment. The category gap is reducing as we go up in the network up to FC7 (almost 0 gap at FC6 and FC7). In contrast the gap is large at all layers for pose estimation. This indicates separation of the instances' view manifolds where the individual manifolds are not collapsed (This is why as we increase the neighborhood, the category performance stays the same while pose estimation decreases smoothly - See Figure 7.3-right for illustration). The results above consistently imply that the higher layers of CNN (expect FC8 which is task specific), even without any fine-tuning on the dataset, and even without any pose label optimization achieve representations that separate and highly preserve the view manifold structure.

The aforementioned conclusion is also confirmed by the test performance of Linear SVM and Kernel Regression in Figure 7.6, using RGBD and Pascal3d+ datasets. In this experiment, the models are learned in train-split and the plots generated using test-split. Figure 7.6 clearly shows the conflict in the representation of the pre-trained network (categorization increases and pose estimation

decreases). Linear separability of category is almost monotonically increasing up to FC6. Linear separability in FC7 and FC8 is worse, which is expected as they are task specific (no fine-tuning). Surprisingly Pool1 features perform very bad, despite being the most general features (typically they show Gabor like features and color blobs). In contrast, for pose estimation, the performance increases as we go lower in the network up to Conv4 and then slightly decreases. This confirms our hypothesis that lower layers offer better feature encoding for pose estimation. It seems that Pool5 provides feature encoding that offer the best compromise in performance, which indicates that it is the best in compromising between the linear separation of categories and the preservation of the view-manifold structure.

Surprisingly, the pose estimation results do not drop dramatically in FC6 and FC7. We can still estimate the pose (with accuracy around 63%) from the representation at these layers, even without any training on pose labels. This highly suggests that the network preserves the view manifold structure to some degree. For examples taking the accuracy as probability at layer FC6, we can vaguely conclude that 90% of the manifolds are linearly separable and 65% are pose preserved (we are somewhere between hypotheses 1 and 2 at this layer).

Tables 7.1 and 7.2 show the quantitative results of our models, compared to baselines, on the RGBD Dataset and Pascal3D+. Table 7.2 shows our quantitative results compared against two previous methods [Zhang et al., 2013] and [Xiang et al., 2014], using the two metrics $< 45^\circ$ and $< 22.5^\circ$ These two metrics < 22.5 and < 45 are defined in [Xiang et al., 2014] as the percentages of test samples that satisfy $AE < 22.5^\circ$ and $AE < 45^\circ$, respectively, where the Absolute Error $AE = |EstimatedAngle - GroundTruth|$. The AAAI pose metric is

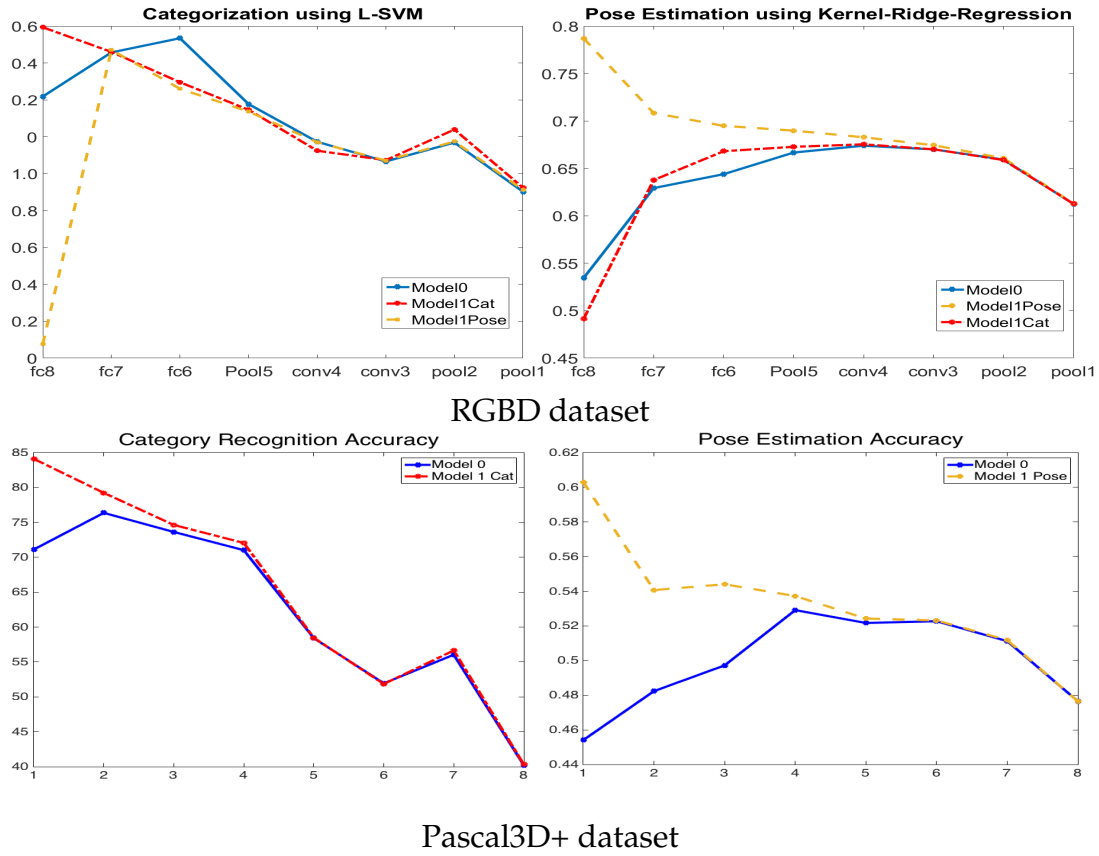


Figure 7.6: Test performance of linear SVM category classification over the layers of different models (Left), and pose regression (Right). For RGBD dataset (Top) and for Pascal3D+ dataset (bottom).

defined as $\Delta(\theta_i, \theta_j) = min(|\theta_i - \theta_j|, 2\pi - |\theta_i - \theta_j|)/\pi$. It is important to note that the comparison with [Xiang et al., 2014] is unfair because they solve for detection and pose simultaneously while we solve for categorization and pose estimation.

Approach	**Categorization** %	**Pose** %
HOG (SVM/Kernel Regression)	80.26	27.95 (AAAI)
Model0 (SVM/Kernel Regression) on conv4	58.64	67.39 (AAAI)
Model0 (SVM/Kernel Regression) on FC6	86.71	64.39 (AAAI)
Model1	89.63	81.21 (AAAI), 69.58 ($<$ 22.5), 81.09 ($<$ 45)

Table 7.1: RGBD Dataset Results for HOG, Model0 and Model1.

Approach	**Categorization % FC6/FC7/FC8**	**Pose (AAAI metric %) FC6/FC7/FC8**	**Pose (AE%)**
Model0 (SVM/Kernel Regression)	73.64/76.38/71.13	49.72/48.24/45.41	
Model1 (SVM/Kernel Regression)	74.65/79.25/84.12	54.41/54.07/60.31	
Model0 NN	60.05/69.89/61.26	61.11/61.38/60.32	
Model1 NN	73.50/77.30/83.07	65.87/66.07/70.54	
Model1 (final prediction)	84.00	71.60	47.34(<22.5), 61.30 (<45)
[Zhang et al., 2013]	-	-	44.20 (< 22.5), 59.00 (<45)
[Xiang et al., 2014]	-	-	15.6 (<22.5), 18.7 (<45)

Table 7.2: Pascal3D+ Performance computed for Model0 and Model1 using different classification techniques. Comparsion indicates that Model1 outperforms the baselines from [Zhang et al., 2013] and [Xiang et al., 2014] using the two metrics $< 45^\circ$ and $< 22.5^\circ$. Model1 here outperforms both baselines, despite the unfair comparison (see text).

7.6 Effect of Transfer Learning

In order to study the effect of fine-tuning the network (transfer learning to a new dataset) on the representation we trained the following model (denoted as Model1). This architecture consists of two parallel CNNs: one with category output nodes (Model1-Cat), and one with binned pose output nodes (Model1-Pose). We used 34 and 11 category nodes for RGBD and Pascal3D datasets respectively; while we used 16 pose nodes for both datasets). The parameters of both CNNs were initialized by Model0 parameters up to FC7. The parameters connecting FC7 to the output nodes are randomly initialized on both networks and they are fine-tuned by minimizing the categorization loss for Model1-Cat and the pose loss for Model1-pose. The purpose of these architectures is to study the effect of fine-tuning when the category and pose are independently optimized.

We applied all the measures described in Sec 7.4 to understand how the view manifolds will be affected after such tuning. The questions are: To what

degree optimizing on category should damage the ability of the network to encode view manifolds. On the other hand, how optimizing on pose should enhance that ability. Model1-Cat indicates the effect of optimizing on category, while Model1-Pose indicates the effect of optimizing on pose.

Figure 7.4 shows the five view manifold measures for the different layers of Model1(Cat/Pose), in comparison with Model0. In terms of data spread, from Figure 7.4a shows that the spread at FC8 has doubled after fine tuning on pose (Model1-Pose). Figure 7.4b shows the fine tuning on category (Model1-Cat) caused the view manifold subspace dimensionality to significantly reduce to 1, where it became totally view invariant. Optimizing on pose slightly enlarged the subspace dimensionality (*i.e,* become better) at FC8 and FC7. Figure 7.4c clearly shows the significant improvement achieved by fine tuning on pose, where the alignment of FC8 jumped to close to 0.9 from about 0.78, while fine tuning on category reduces the alignment of FC8 to close to 0.65. Similar behavior is also apparent in the KPLS ratio for FC8 and FC7 (sup-mat).

One very surprising result is that optimizing on pose makes the pose KTA alignment worse at the lower layers, while optimizing on category makes the pose alignment better compared to model0. In fact, although optimizing on pose significantly helps aligning FC8 with pose labels, Pool5 still achieves the best KTA alignment and the least regression reconstruction error. The regression reconstruction error in Figure 7.4d clearly shows significant improvement in the representation of FC8 and FC7 to preserve the view manifold. One surprising finding from these plots is that the representation of FC6 becomes worse after fine tuning for both pose and category. Figure 7.4e indicates that the deformation of the view manifold is reduced as a result of fine tuning on

pose (larger rcond number), while it increases as a result of fine tuning on category.

On the global object-view manifold structure, we notice from Figures 7.6 some intuitive behavior at FC8. Basically optimizing on pose reduces the linear separability and increases the view manifold preservation (moves the representation towards hypothesis 0). In contrast, optimizing the category significantly improves the linear separability at FC8, however, interestingly, it only slightly reduces the pose estimation performance to be slightly less than 50%. Combining this conclusion with the observation from Figure 7.4b, that the view manifold subspace dimensionality reduces to 1, this implies that optimizing on category collapses the view manifolds to a line, but they are not totally degenerate. What is less obvious is the effect of fine tuning on the lower layers than FC8. Surprisingly, optimizing on pose did not affect the linear separability of FC7. Another very interesting observation is that optimizing on category actually improves the pose estimation slightly at the FC7, FC6, and Pool5; and did not reduce it at lower layers. This implies that fine tuning by optimizing on category only improved the internal view manifold preservation at the network, even without any pose labels.

7.7 More Measurements and Results

In this section, we define more measurements such as HSIC, KPLS-Norm Ratio and TPS-nonPolynomial. Moreover, we elaborate on already measurements in Section 7.4.1. In this section, we use the same notations and definitions stated in Section 7.4.1.

Hilbert-Schmidt Independence Criterion:

HSIC Gretton et al. [2005] is a convergence measure between two kernels (affinity matrices). We use HSIC to measures the correlation between the two sets of points (the set of points on the view-manifold and set of points on the circle) based on the spatial structure of the points in each set, by measuring how far they are both extracted from the same distribution. Empirically HSIC is approximated by $HSIC(A,B) = \frac{1}{(n-1)^2} trace(A * H * B * H)$, where A and B are two affinity matrices, $H = I_n - \frac{1}{(n)}\mathbf{1}_{n\times n}$, $\mathbf{I}$ is the unit matrix and $\mathbf{1}$ is matrix of all ones. The matrix H centerlizes the data in the feature space.

KPLS-based measurements:

As mentioned in Section 2.2, Kernel Partial Least Squares (KPLS) iteratively extracts a set of principal components of the input kernel that are most correlated with the output. Unlike KPCA extracts the principal components (PCs) of the kernel of the input data to maximize the variance of the output space, KPLS extracts the PCs of the kernel of the input data that maximize the correlation with the output data. We use KPLS to map the affinity matrix of the transformed view-manifold (*v*iew-kernel) to the circle affinity matrix (*c*ircle-kernel). Following the convention in Section 7.4.1, let the view-kernel is denoted by $\mathbf{K}^l$, and the circle-kernel is denoted by $\mathbf{K}^\circ$ (*T*he subscript n is removed to simplify the notation). We limit the number of extracted PCs to d , where $d \ll N$ and N is the dimensionality of the input kernel (in this work, we use $d = 5$). More specifically, KPLS maps the rows of $\mathbf{K}^l$ to the rows of $\mathbf{K}^\circ$. So that

$$\hat{\mathbf{K}}^\circ = \mathbf{G_0 U}(\mathbf{T}^\top \mathbf{G_0 U})^{-1}\mathbf{T}^\top \mathbf{K}^l \tag{7.1}$$

Where the set of extracted PCs are the columns of the matrix $T_{N\times d}$, $U_{N\times d}$ is auxiliary matrix, and the Gram-matrix $\mathbf{G_0}$ is defined by

$$\mathbf{G}_0 = \frac{\mathbf{K}^l {\mathbf{K}^l}^\top}{bb^\top} \tag{7.2}$$

Where $b \in \mathbb{R}^N$, so that $b(i)$ is the Frobenius norm of the i-th row of $\mathbf{K}^l$. Based on the mapping in Eq 7.1, we extract two measurements:

First: **KPLS-Regression Error (δ)** which measures geometric deformation of the generated output image of view-kernel in the circle-kernel space ($\hat{\mathbf{K}}^\circ$ with respect to the the circle-kernel ($\mathbf{K}^\circ$) and the). One choice for measuring this is

$$\delta(\hat{\mathbf{K}}^\circ, \mathbf{K}^\circ) = 1 - KTA(\hat{\mathbf{K}}^\circ, \mathbf{K}^\circ)$$

, where KTA stands for Kernel Target Alignment (stated in Section 7.4.1). The Regression error measures the reconstruction error of the circle-kernel from the view-kernel.

Second, **KPLS-NormK Ratio ($\frac{\|G_d\|_F}{\|G_0\|_F}$)** measures the residual energy after extracting the first d-PC's. Where G_d is the residual of G_0 after d-iterations. The intuition behind this measure is that the larger the ratio $\frac{\|G_d\|_F}{\|G_0\|_F}$, this means that the view-manifold has more than d-PC's correlated with the circle-kernel.

While KPLS-regression Error is self-explanatory (this measure presented in Section 7.4.1), using the two KPLS measurements together gives more precise view on the correlation between the view-manifold and the circle-manifold. From Figure 7.7b, KPLS-Norm Ratio supports the observation that we noted in Section 7.4.1, from Figure 7.7a, that the lower layers in Model0 are more correlated to the circle-manifold than the higher layers. Except for Pool5, which encodes maximum correlation between the view-manifold and the circle-manifold.

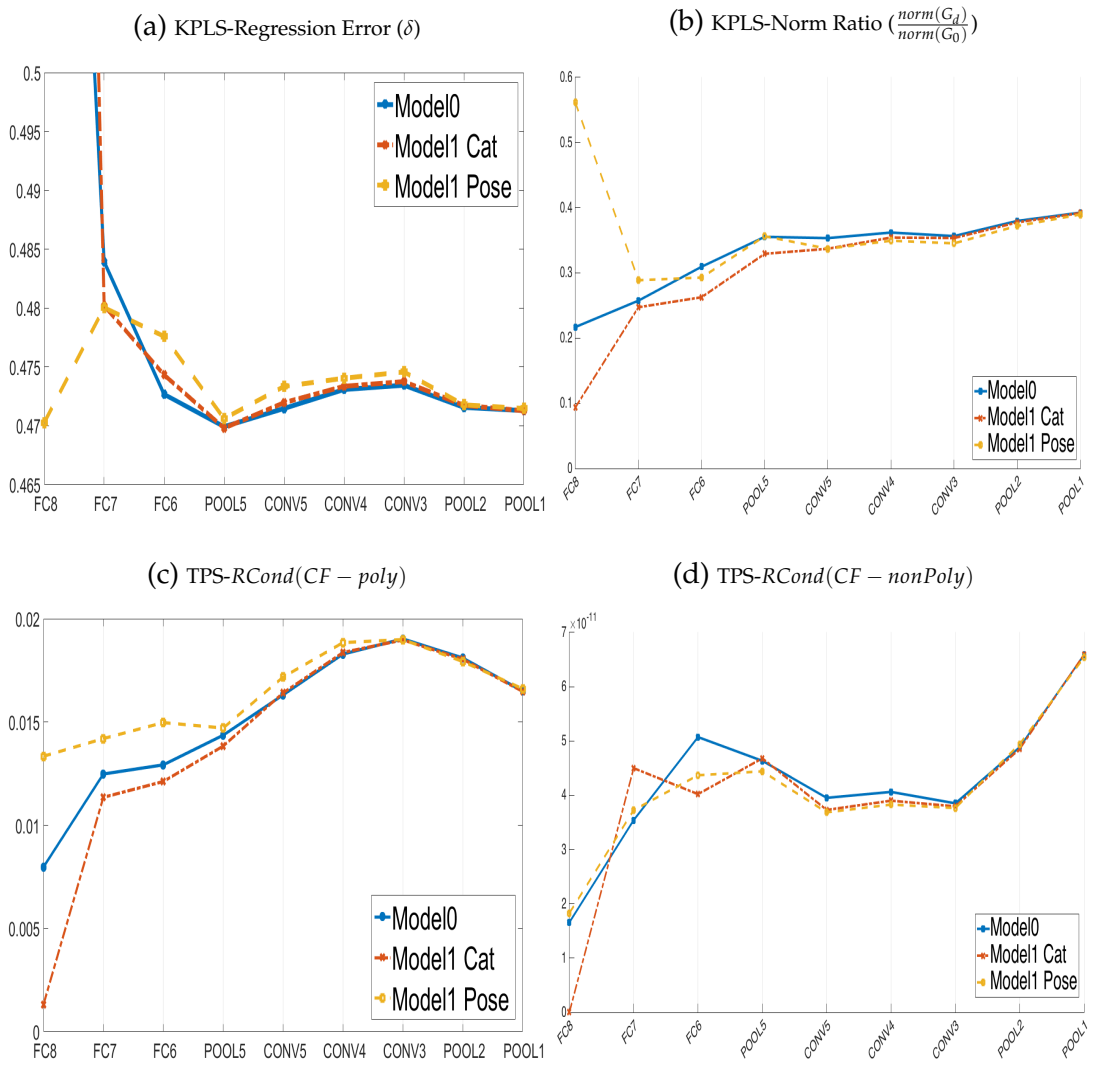


Figure 7.7: Measurement analysis for the view-manifold in RGBD dataset based on features extracted from different layers of several CNN models. Every figure shows single measurement. Multiple lines is for different CNN model. X-axis is labeled by the layers.

TPS-nonlinearity measure:

In this measure we learn a regularized Thin Plate Spline (TPS) non-linear mapping Duchon [1977] between the unit circle manifold and each manifold $\mathcal{M}^k$. The mapping function (γ) can be written as

$$\gamma^k(\mathbf{x}) = \mathbf{C}^k \cdot \psi(\mathbf{x}),$$

where $\mathbf{C}_{d\times(N+e+1)}$ is the mapping matrix, $e = 2$, and the vector $\psi(x) = [\phi(|x - z_1|) \cdots \phi(|x - z_M|), 1, x^T]^T$ represents a nonlinear kernel map from the conceptual representation to a kernel induced space. The thin plate spline is defined as: $\phi(r) = r^3$ and $\{z_i\}_{i=1}^M$ are the set of center points. The solution for $\mathbf{C}^k$ can be obtained by directly solving the linear system:

$$\begin{pmatrix} \mathbf{K}^l + \lambda\mathbf{I} & \mathbf{P}_x \\ \mathbf{P}_t^T & \mathbf{0}_{(e+1)\times(e+1)} \end{pmatrix}_k \mathbf{C}^{k^T} = \begin{pmatrix} \mathbf{A}_k \\ \mathbf{0}_{(e+1)\times d} \end{pmatrix}, \tag{7.3}$$

$\mathbf{A}$, $\mathbf{P}_x$ and $\mathbf{P}_t$ are defined for the $k - th$ set of object images as: $\mathbf{A}$ is a $N_k \times M$ matrix with $\mathbf{K}^l_{ij} = \phi(|x_i^k - z_j|), i = 1, \cdots, N_k, j = 1, \cdots, M, \mathbf{P}_x$ is a $N_k \times (e + 1)$ matrix with i-th row $[1, \mathbf{x}_i^{k^T}]$, $\mathbf{P}_t$ is $M \times (e + 1)$ matrix with i-th row $[1, \mathbf{z}_i^T]$. A_k is a $N_k \times d$ matrix containing the set of images for manifold $\mathcal{M}^k$, *i.e.* $\mathbf{A}_k = [\mathbf{y}_1^k, \cdots, \mathbf{y}_{N_k}^k]$. Solution for $\mathbf{C}^k$ is guaranteed under certain conditions on the basic functions used.

The reason for using TPS in particular is that the mapping has two parts, an affine part (linear polynomial) and a nonlinear part. Inquiring into the two parts gives an impression about the mapping, if it is mostly linear or nonlinear. We used the *r*eciprocal-condition number (RCond) of the submatrices of the coefficient matrix that correspond to the affine and the nonlinear part.

While Figure 7.7c shows that the lower layers has more (better) conditioned linear mapping. Figure 7.7d shows that the lower layer has complete stable mapping. This is expected since the lower layers have high dimensionality. At the same time, Figure 7.7d shows that the Convolution layers (Conv 3,4 and 5) have unstable nonlinear mappings. An additional observation is that fine-tuning against the pose labels increases the mapping stability (polynomial and non-polynomial). It is clear in Figure 7.7d that the $TPS - RCond(CF - nonPoly)$ has very small order of values (10^{-11}), therefore, we do not rely on it

in our analysis.

7.8 Conclusions

In this study, we present an in-depth analysis and discussion of the view-invariant properties of CNNs. We proposed a methodology to analyze individual instance's view manifolds, as well as the global object-view manifold. We applied the methodology on a pre-trained CNN, as well as two fine-tuned CNNs, one optimized for category and one for pose. We performed the analysis based on two multi view datasets (RGBD and Pascal3D+). Applications on both datasets give consistent conclusions.

Based on the proposed methodology and the datasets, we analyzed the layers of the pre-trained and fine-tuned CNNs. There are several findings from our analysis that are detailed throughout the chapter, some of them are intuitive and some are surprising. We find that a pre-trained network captures representations that highly preserve the manifold structure at most of the network layers, including the fully connected layers, except the final layer. Although the model is pre-trained on ImageNet, not a densely sampled multi-view dataset, still, the layers have the capacity to encode view manifold structure. It is clear from the analysis that, except of the last layer, the representation tries to achieve view invariance by separating individual instances' view manifolds while preserving them, instead of collapsing the view manifolds to degenerate representations. This is violated at the last layer which enforces view invariance.

*O*verall, our analysis using linear SVM, kernel regression, KNN, combined

with the manifold analysis, makes us believe that CNN is a model that simulate the manifold flattening hypothesis of DiCarlo and Cox [2007] even without training on multi-view dataset and without involving pose labels in the objective's loss.

Another interesting finding is that Pool 5 offers a feature space where the manifold structure is still preserved to the best degree. Pool 5 shows better representation for the view-manifold than early layers like Pool1. We hypothesize that this is because Pool5 has better translation and rotation invariant properties, which enhance the representation of the view manifold encoding.

We also showed the effect of fine-tuning the network on multi-view datasets, which can achieve very good pose estimation performance. In this work, we only studied the effect of independent pose and category loss optimization. Optimizing on category achieves view invariance at the very last fully connected layers; interestingly it enhances the viewpoint preservation at earlier layers. We also find that fine-tuning mainly affects the higher layers and rarely affects the lower layers.

In this work our goal is not to propose any new architecture or algorithm to compete with the state of the art in pose estimation. However, the proposed methodology can be used to guide deep network design for solving several tasks. To show that and based on the analysis and the conclusions of this chapter, we introduced and studied in [Elhoseiny et al., 2015] several variants of CNN architectures for joint learning of pose and category, which outperform the state of the art .

Chapter 8
Conclusion

This dissertation highlights the power of manifolds of images for building a robust data modeling. For leveraging the image manifolds, the dissertation proposed several extensions to the Homomorphic Manifold Analysis (HMA) framework and offered a probabilistic generalization to HMA. The dissertation also adopted image manifold analysis to introduce deeper understanding of CNN.

The dissertation proposed Manifold Kernel Partial Least Squares (MKPLS) framework to model the variations between several image manifolds. MKPLS is an extension to HMA, so that it parameterizes the manifolds using Gaussian Radial Basis Functions (Gaussian RBF), and it customizes Kernel Partial Least Squares (KPLS) for embedding the manifold parameterizations into low-dimensional latent space. MKPLS is intended to discriminate between different manifolds. The novelty here is applying KPLS on instance manifolds, based on its parameterization, to reach a latent space where each manifold is represented by a point. This is in contrast to traditional usage of KPLS on data points, where each image is a point. The dissertation applied MKPLS on visual speech recognition to recognize the spoken word in utterance footage and to identify speaker. By experimenting on two public databases, we showed the results for three different settings: speaker independent, speaker dependent, and speaker semi-dependent. Our approach outperforms for the speaker

semi-dependent setting by at least 15% of the baseline, and competes in the other two settings.

Furthermore, The dissertation considered the problem of modeling the combined object-viewpoint manifold. Since multiple viewpoints of an object lie on an intrinsic low-dimensional manifold in the input feature space. We observed that different objects captured from the same set of viewpoints have manifolds with a common topology. Based on this observation, the dissertation proposed a nonlinear generative model that represent the images in the feature space as a function of two factor of variations: the viewpoint and the object-category. The dissertation designed a novel framework, that learns two latent spaces to encode the variations in these two factors, the viewpoint and the category inter and intra-variations. This framework extends the MKPLS framework from being manifold-based to be image-based. This is done by learning a Gaussian-RBF regression function from the style latent space to the feature space. The advantage of this modeling is that the inference procedure is moved from the very high-dimensional coefficient mapping space to two low-dimensional pose and category spaces, which makes the inference more accurate and computationally easier. We also incorporate this model into a hierarchical structure to deal with large intra-class variation. For inference, the dissertation compared the performance of two continuous-domain and iterative algorithms one of them based on sampling and the other one based on gradient descent. These two algorithms explore the style space and the viewpoint space looking for the best match for a given test image. The dissertation empirically validated the model by applying the proposed framework on multiple challenging multi-view datasets. We compare our results with the state-of-the-art and present our increased category recognition and pose estimation accuracy.

Despite the advantages of the proposed continuous-domain iterative algorithm for inference, it is computationally expensive. This limits its applicability in real-life computer vision system. Therefore, the dissertation proposed an efficient feedforward framework that can untangle such a complex object-view manifold, and achieve a model that separates a view-invariant category representation, from category-invariant pose representation. The proposed feedforward framework is inspired by the hypothesis introduced by DiCarlo and Cox [2007] for the human vision system. The dissertation proposed several approaches to speed up the inference phase. They are based mainly on discretizing one, and only one, of the two learned latent spaces. Depending on which latent space selected for discretization, we introduced a framework that is based on projecting into the low-dimensional space, in which we perform the inference. More specifically, the dissertation compared using three different projectors: view-specific object-invariant, instance-specific view-invariant, or category-specific view-invariant projectors. The dissertation concluded that the view-specific object-invariant based framework is the best one. In this case we discretized the viewpoint latent space and introduce view-specific projectors that map the test images into the low-dimensional style space, in which we perform the inference. By experimenting on three multi-view dataset, we showed the potentials of our proposed approach. The dissertation showed empirically that the proposed framework is not only more efficient but also more accurate than the previous framework for multi-view object recognition and pose estimation. The proposed framework outperformed the reported state-of-the-art approaches by significant margin.

In addition, the dissertation proposed a probabilistic bi-nonlinear generative model based on Gaussian Processes (GP), to find a unified content latent

space and discriminate the styles, for several computer vision applications. In this work, we proposed a novel multi-factor probabilistic generative model, based on Gaussian processes, for modeling the intrinsic variations of multi instance datasets. In order to do this, the dissertation derived closed form for the likelihood of points in the input space given points in the content and style spaces. Unlike the aforementioned generative model, which manually sets the content (viewpoint) space and learns only the style space. The proposed framework in this part of the dissertation introduced iterative algorithm, on top of Gaussian process latent variable model (GPLVM), that embeds the training data into two induced style and content latent spaces. This modification helps to conquer several computer vision applications such as human motion tracking and recognition. To validate the framework, we applied the approach on a synthetic dataset. To show the efficiency of the proposed framework, the dissertation conducted experiments on several public dataset for visual human motion. We showed that our approach is able to capture the actual motion topology in the CMU MOCAP dataset. Moreover, it learns style space that encodes the variations among different motion styles in this dataset. To show the applicability of the proposed framework in many computer vision applications: the dissertation showed preliminary results on human motion analysis dataset, multi-view object recognition dataset and visual speech recognition dataset.

In the last part of the dissertation, we present an in-depth analysis and discussion of the view-invariant properties of CNNs. We proposed a methodology to analyze individual instance's view manifolds, as well as the global object-view manifold. We applied the methodology on a pre-trained CNN, as well as two fine-tuned CNNs, one optimized for category and one for pose.

We performed the analysis based on two multi-view datasets (RGBD and Pascal3D+). Based on the proposed methodology and the datasets, the dissertation constitutes set of observations and conclusions. First, a pre-trained network captures representations that highly preserve the manifold structure at most of the network layers, including the fully connected layers, except the final layer. The analysis using linear SVM, kernel regression, KNN, combined with the manifold analysis, makes us believe that CNN is a model that simulate the manifold flattening hypothesis of DiCarlo and Cox [2007] even without training on multi-view dataset and without involving pose labels in the objective's loss. We also showed the effect of fine-tuning the network on multi-view datasets, which can achieve very good pose estimation performance. While optimizing on category achieves view invariance at the very last fully connected layers; interestingly it enhances the viewpoint preservation at earlier layers, but not vise versa. Another interesting finding is that Pool 5 offers a feature space where the manifold structure is still preserved to the best degree. Pool 5 shows better representation for the view-manifold than early layers like Pool1. The proposed methodology can be used to guide deep network design for solving several tasks. To show that, we introduced and studied in [Elhoseiny et al., 2015] several variants of CNN architectures for joint learning of pose and category, which outperform the state of the art .

Appendix A
Synthetic Dataset for Understanding CNN

A.1 Motivation

In the work presented in Chapter 7 for monitoring the view-invariance in CNN, we have explored many different measurements and filter them out to use only those that expose the correct properties of the view-manifolds. Besides the intuitive reasoning that we provided for choosing the measurements, in this section, we show empirical results to quantify efficiency of the chosen measurements.

To this end, we synthesized a set of well designed view-manifolds. Analyzing these manifolds is intended to identify the robust and informative set of measurements to be used in further analysis. To be qualified for comparing different manifolds, the synthesized manifolds is designed to encode interesting properties of any view-manifold such as:

- Dimensionality (of the Euclidean space where the manifold lives)
- Sparsity of the manifold
- Smoothness of the manifold
- Deformation of the manifold w.r.t the *view-circle*
- Variance of data-points

Recall, The view-circle is a view-manifold, where all the viewpoints form a perfect circle and the object is assumed to be located at the center of this circle. In the rest of this section, we list detailed description of the synthetic manifolds. Then, we use them to analyze the selected measurements.

(a) Sinosoidal Surface (b) Circle projected on S_2 (c) Circle projected on S_2

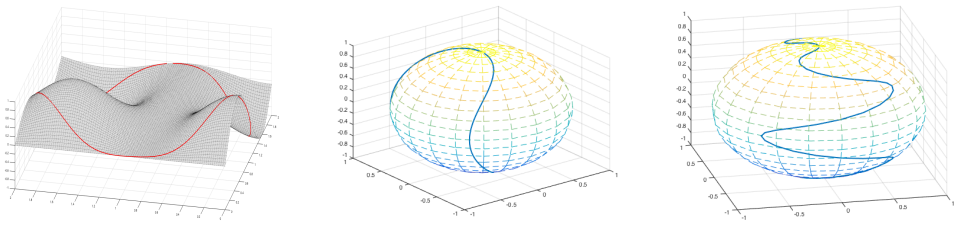


Figure A.1: Manifold Visualization

A.2 Dataset Description

As in Figure A.1, manifolds in this dataset can be categories as:

- Circle Orthogonally projected to high-dimensional subspaces (Manifold sets 1 and 2)
- Unit circle projected to a nonlinear surface (manifold 3)
- Unit circle projected to $3D$-Sphere with radius r (S_2^r) (sets 4 an 5)
- Nonlinear smooth curve projected on S_2^r (set 6)
- Discontinuous smooth curve projected on S_2^r (set 7)
- Random manifolds (sets 8 and 9)
- Collapsed manifolds in a single point or very small region (set 10).

The manifolds are described using the dimensionality (d), sparsity ($s = \frac{n}{d}$) and n, the number of points representing the view-manifold, smoothness, deformation w.r.t the view-circle.

Let the view-manifold be parameterized by the single dimensional variable. Let S is the two dimensional representation of the unit circle. $S = \{(cos(t), sin(t))|t = \{0, \frac{2\pi}{n}, \frac{4\pi}{n}, ..., \frac{2(n-1)\pi}{n}\}\}$. For each view-manifold ($\mathcal{M}$), we generated n points in a d-Dim space.

- **Perfect view-circle in high-dimensional space**
  - Manifold 1: $n = 100$, $d \in \{10, 300, 600, 900, 1200, 1500, 1800\}$, therefore, the sparsity varies from very dense ($s = 10$) to very sparse ($s = 1/20$)
  - Manifold 2: $d = 500$, $n \in \{50, 150, 250, 350, 450, 550, 650, 750\}$, therefore, the manifold varies from very sparse ($s = 1/10$) to dense ($s = 1.5$)

- **View-circle projected nonlinearly to Sinusoidal Surface**. The manifold has $n = 100$ points and live in $d = 3$-Dim space, so it is very dense $s = 33.33$ To project the view-circle on this surface we follow these steps:
  - Let fn be the projection function on the surface,

$$fn(x, y) = sin(3x)cos(2y)^2$$

  - The projected manifold Z is defined by

$$Z = \{(x, y, fn(x, y))|(x, y) \in S\}$$

  - Manifold 3 represents this type of manifolds in our dataset.

- **Dense view-circle projected nonlinearly to S_2^r, with $r \in \{1, 50, 100, 150\}$, $d = 3$, $n = 100$ ($s = 33.33$)**

  To project the view-circle on S_2^r, we use the following projection function

  $$f(\theta, \phi) = (sin(\phi)cos(\theta), sin(\phi)sin(\theta), cos(\phi))$$

  Where

  $$\theta \in \{0, \frac{2\pi}{n}, \frac{4\pi}{n}, ..., \frac{2(n-1)\pi}{n}\}$$

  - Manifold 4: Slightly deformed manifold, Figure A.1b

    $$\phi = \frac{\pi}{4}sin(\theta) + \frac{\pi}{2}; \forall\theta$$

  - Manifold 5: Slightly deformed manifold with added Gaussian noise with $\mu = 0$ mean $\sigma = 0.01$. θ and ϕ as in Manifold 4.

  - Manifold 6: Highly deformed manifold, Figure A.1c

    $$\phi = \frac{\pi}{4}sin(5\theta) + \frac{\pi}{2}; \forall\theta$$

  - Manifold 7: Highly deformed and broken/discontinuous manifold.

    $$\phi = \frac{\pi}{4}tan(0.75\theta) + \frac{\pi}{2}; \forall\theta$$

- **Random manifold with independent dimensions**

  - Manifold 8: Uniform random points with $d \in \{10, 100, 500, 1000, 4000\}$, $n = 100$, therefore, the sparsity varies from very dense ($s = 10$) to very sparse ($s = 1/100$)

  - Manifold 9: Normal random points has been generated with $d = 100, n \in \{20, 40, ..., 200\}$, therefore, sparsity varies from very sparse ($s = 1/10$) to dense ($s = 2$)

- **Collapsed Manifold with random noise**
  - Manifold 10: Portion of the points ($m = n/4$), in this manifold, have been generated by Gaussian Random with $\mu = 0$, $\sigma = 0.01$, therefore, the rest are a copied version of this portion $d \in \{10, 100, 500, 1000, 4000\}$, $n = 100$

A.3 Measurement Analysis

Recall, the objective of using the synthetic data is to verify the efficiency of selected measurements. Figure A.2 shows the results of applying the measurements to the synthetic-data. Figure A.2a shows the Nuclear Norm for all manifolds. This figure shows the variability between the manifolds in the variance. For the set of manifolds 4-7, projecting the view-circle onto sphere with different sizes affects the variance of the points. Encoding different Nuclear Norm is subjected to discover the measurements that are sensitive to the data variance.

From Figure A.2b, we can see the effective dimensions for each manifold. Manifolds 1 and 2 have two effective dimensions. Manifolds 3-7 has three effective dimensions. Since the points in Manifolds 8-10 are generated randomly so they have maximum rank.

The kernel alignment measures: KTA (Figure A.2c) and HISIC (Figure A.2d) measure the correlation between the view-manifold and the view-circle. These two figures show significant better alignment of the view-manifold of sets 1-6 than the alignment of the random manifolds. Since Hilbert-Schmidt Independence Criterion (HSIC) Gretton et al. [2005] does not add any information more

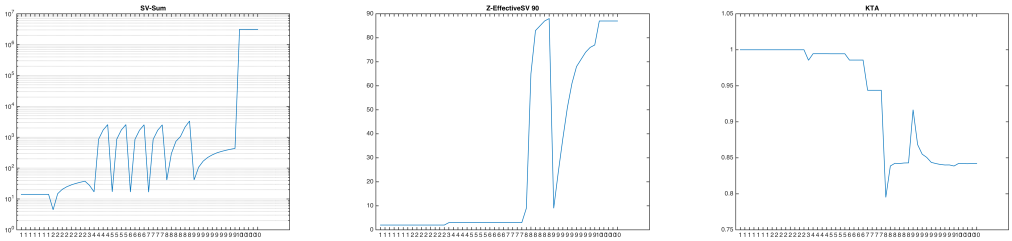
than KTA. We select the KTA measurement because it exposes absolute alignment confidence for the manifolds 1 and 2.

KPLS-regression Error is shown in Figure A.2e. Dispite the vast variability of variance and dimensionality, this measure is consistent and gives small value for all smooth manifold. This measure can also detect the collapsing manifolds, since it gives very large error value.

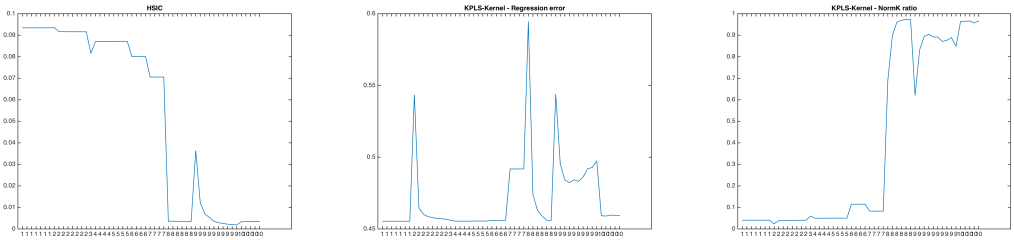
As we mentioned in Section 7.7, that using both measurements KPLS-Regression Error and KPLS-Norm Ratio gives more robust conclusion about the manifold. Figure A.2f shows a clear trend, since it gives significant high values for random manifolds. This is because, the subspace of the random points covers the entire space. When $\frac{norm(G_d)}{norm(G_0)} \equiv 1$, this means that the firt d components extracted from G_0 are far from being principal. If they are pricipal components, they would change the energy of the matrix G significantly. On the other side, the Effecive dimensionality of the smooth manifolds 1-6 is $D \leq 3$, which make the limit $d > D$. That is why the ratio $\frac{norm(G_d)}{norm(G_0)} \ll$ because we have extracted all the pricipal components of those manifolds. That is why KPLS-Regression Error for these manifolds is very small.

TPS-lineairty measure ($TPS - RCond(CF - Poly)$) scores on the stability of the polynomial mapping from the points on the view-circle and the points on the view-manifold. Figure A.2g shows perfect scoring for Manifolds 1 and 2. Combining this figure with Figure A.2h gives a complete impression about the mapping stability (Polynomial and Non-Polynomial). However, the range of the values of TPS-nonlinearity measure ($TPS - RCond(CF - nonPoly)$) is in $BigO(10^{-8})$, which decrease its robustness.

(a) Nuclear Norm (b) Effective 90% SV's (c) KTA



(d) HSIC (e) KPLS-Regression error (f) KPLS-$\frac{norm(G_d)}{norm(G_0)}$



(g) TPS-$RCond(CF-poly)$ (h) TPS-$RCond(CF-nonPoly)$

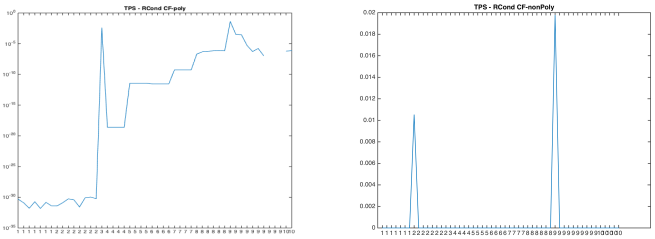


Figure A.2: Measurement analysis for the synthetic manifolds. Every figure shows single measurement. X-axis is labeled by the manifold category number.

Bibliography

Pulkit Agrawal, Ross Girshick, and Jitendra Malik. Analyzing the performance of multilayer neural networks for object recognition. In *Computer Vision–ECCV 2014*, pages 329–344. Springer, 2014. 128

M. Aizerman, E. Braverman, and L. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964. 21

Andrzej P. Ruszczynski. *Nonlinear Optimization, Volume 13*. Princeton University Press, 2006. 60

Sherif Azary and Andreas Savakis. Grassmannian sparse representations and motion depth surfaces for 3d action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 492–499. 2013. 3

A. Bakry, T. Elgaaly, M. Elhoseiny, , and A. Elgammal. Joint object recognition and pose estimation using a nonlinear view-invariant latent generative model. *WACV*, 2016. 13

A. Bakry and A. Elgammal. Mkpls: Manifold kernel partial least squares for lipreading and speaker identification. *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 684–691, June 2013. 13, 50

A. Bakry and A. Elgammal. Manifold-Kernels Comparison in MKPLS for Visual Speech Recognition. *ArXiv e-prints*, jan 2016. 13

Amr Bakry and Ahmed Elgammal. Untangling object-view manifold for multiview recognition and pose estimation. *ECCV 2014*, pages 434–449, 2014. 13, 50, 53, 129

Amr Bakry, Mohamed Elhoseiny, Tarek El-Gaaly, and Ahmed Elgammal. Digging deep into the layers of cnns: In search of how cnns achieve view invariance. *arXiv preprint arXiv:1508.01983*, 2015. 13

KP Bennett and MJ Embrechts. An optimization perspective on kernel partial least squares regression. *Nato Science Series, Sub-Series III: computer and System Sciences*, 190:227–249, 2003. 19, 20

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc., 2006. ISBN 0387310738. 60

Liefeng Bo and Cristian Sminchisescu. Twin gaussian processes for structured prediction. *International Journal of Computer Vision*, 87(1-2):28–52, 2010. 113

Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *CoRR*, abs/1405.3531, 2014. URL `http://arxiv.org/abs/1405.3531`. 124, 128

Stephen Cox, Richard Harvey, and Yuxuan Lan. The challenge of multi-speaker lip-reading. *International Conference on Auditory-Visual Speech Processing*, 2008. 40

CM Cyr and BB Kimia. A similarity-based aspect-graph approach to 3D object recognition. *International Journal of Computer Vision*, 2004. URL `http://link.springer.com/article/10.1023/B%3AVISI.0000013088.59081.4c`. 53

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. *CVPR*, 2005. 8, 64, 91, 92

Andreas Damianou, Carl Ek, Michalis Titsias, and Neil Lawrence. Manifold relevance determination. *arXiv preprint arXiv:1206.4610*, 2012. 103

Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A Multilinear Singular Value Decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1253–1278, jan 2000. 101

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009. 1, 3, 132

James J DiCarlo and David D Cox. Untangling invariant object recognition. *Trends in cognitive sciences*, 11(8):333–341, 2007. xvi, 83, 125, 126, 129, 153, 156, 158

James J DiCarlo, Davide Zoccolan, and Nicole C Rust. How does the brain solve visual object recognition? *Neuron*, 73(3):415–434, 2012. 76

Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. *arXiv preprint arXiv:1310.1531*, 2013. 128

David L Donoho and Carrie Grimes. Image manifolds which are isometric to euclidean space. *Journal of mathematical imaging and vision*, 23(1):5–24, 2005. 4

Jean Duchon. Splines minimizing rotation-invariant semi-norms in sobolev spaces. In *Constructive theory of functions of several variables*, pages 85–100. Springer, 1977. 137, 150

R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley, 2001. 19, 62

Alan Edelman, Tomás a. Arias, and Steven T. Smith. The Geometry of Algorithms with Orthogonality Constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2):303–353, jan 1998. 39, 88

Carl Henrik Ek, Jon Rihan, Philip HS Torr, Grégory Rogez, and Neil D Lawrence. Ambiguity modeling in latent spaces. In *Machine Learning for Multimodal Interaction*, pages 62–73. Springer, 2008a. 103

Carl Henrik Ek, Philip HS Torr, and Neil D Lawrence. Gaussian process latent variable models for human pose estimation. In *Machine learning for multimodal interaction*, pages 132–143. Springer, 2008b. 103

A. Elgammal and CS Lee. Separating style and content on a nonlinear manifold. *IEEE Conference on Computer Vision and Pattern Recognition*, 1:478–485, 2004a. 3, 14, 15, 16, 17, 26, 27, 57, 101

Ahmed Elgammal and Chan-Su Lee. Inferring 3d body pose from silhouettes using activity manifold learning. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–681. IEEE, 2004b. 15

Ahmed Elgammal and Chan-Su Lee. Tracking people on a torus. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(3):520–538, 2009. 15

Ahmed Elgammal and Chan-Su Lee. Homeomorphic manifold analysis (hma): Generalized separation of style and content on manifolds. *Image and Vision Computing*, 31(4):291–310, 2013. 5, 14

Mohamed Elhoseiny, Tarek El-Gaaly, Amr Bakry, and Ahmed Elgammal. Convolutional models for joint object categorization and pose estimation. *arXiv preprint arXiv:1511.05175*, 2015. 153, 158

Mark Everingham, Luc Van Gool, C. K. I. Williams, J. Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision (IJCV)*, 2010. 132

Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2010. 74

Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61(1):55–79, 2005. 74

Oren Freifeld and Michael J Black. Lie bodies: A manifold representation of 3d human shape. In *Computer Vision–ECCV 2012*, pages 1–14. Springer, 2012. 3

Yun Fu and Xi Zhou. Lipreading by locality discriminant graph. *IEEE International Conference on Image Processing*, pages 325–328, 2007. 28

Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, abs/1311.2524, 2013. URL `http://arxiv.org/abs/1311.2524`. 124

Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In *Proceedings of the 16th International Conference on Algorithmic Learning Theory*, pages 63–77. Springer-Verlag, Berlin, Heidelberg, 2005. 136, 148, 163

WJEL Grimson and T Lozano-Perez. Recognition and localization of overlapping parts from sparse data in two and three dimensions. In *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, volume 2, pages 61–66. IEEE, 1985. 73

J. Ham, Lee Daniel, and Lawrence Saul. Semisupervised alignment of manifolds. *Proceedings of the Annual Conference on Uncertainty in Artificial Intelligence*, 2005. 17

MT Harandi and Conrad Sanderson. Graph embedding discriminant analysis on Grassmannian manifolds for improved image set matching. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2705–2712, June 2011. 39, 57

TJ Hazen, Kate Saenko, Chia-hao La, and JR Glass. A segment-based audio-visual speech recognizer: Data collection, development, and initial experiments. *Proceedings of the 6th international conference on Multimodal interfaces. ACM*, 2004. 40

Roger A. Horn. *Matrix Analysis*. Cambridge university press, 2013. 135

Kevin Jarrett, Koray Kavukcuoglu, and Yann Lecun. What is the best multi-stage architecture for object recognition? 2009. 127

Koray Kavukcuoglu, Pierre Sermanet, Y-Lan Boureau, Karol Gregor, Michaël Mathieu, and Yann L Cun. Learning convolutional feature hierarchies for visual recognition. In *Advances in neural information processing systems*, pages 1090–1098. Curran Associates, Inc., 2010. 127

G. S. Kimeldorf and G. Wahba. A correspondence between bayesian estimation on stochastic processes and smoothing by splines. *The Annals of Mathematical Statistics*, 41:495–502, 1970. 18, 80

Emile H. L. Aarts; Jan Korst. *Simulated annealing and Boltzmann machines: a stochastic approach to combinatorial optimization and neural computing*. Wiley, 1989. 60

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105. 2012. 12, 124, 127, 133

Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *ICRA*, pages 1817–1824. 2011a. 62, 64, 90, 92, 131

Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. A scalable tree-based approach for joint object and pose recognition. In *AAAI*. 2011b. x, 50, 53, 68, 70, 93, 94

Y. Lamdan and H. Wolfson. *Geometric hashing: A general and efficient model-based recognition scheme*. New York University, Department of Computer Science, Courant Institute of Mathematical Sciences, 1988. 73

Lieven De Lathauwer, Bart de Moor, and Joos Vandewalle. A multilinear singular value decomposiiton. *SIAM Journal On Matrix Analysis and Applications*, 21(4):1253–1278, 2000a. 81

Lieven De Lathauwer, Bart de Moor, and Joos Vandewalle. On the best rank-1 and rank-(r1, r2, ..., rn) approximation of higher-order tensors. *SIAM Journal On Matrix Analysis and Applications*, 21(4):1324–1342, 2000b. 55

Neil Lawrence. Probabilistic non-linear principal component analysis with gaussian process latent variable models. *The Journal of Machine Learning Research*, 6:1783–1816, 2005. 10, 99, 100, 101, 117

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 3

Yann LeCun, Fu Jie Huang, and Léon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, CVPR'04, pages 97–104. IEEE Computer Society, Washington, DC, USA, 2004. 127

Bowon Lee and et al. AVICAR: Audio-visual speech corpus in a car environment. *Proc. Int. Conf. Spoken Lang. Process*, 2004. 40

C-S Lee and A Elgammal. Coupled visual and kinematic manifold models for tracking. *International Journal of Computer Vision*, 87(1-2):118–139, 2010. 15

Chan-Su Lee and Ahmed Elgammal. Facial expression analysis using nonlinear decomposable generative models. In *Analysis and Modelling of Faces and Gestures*, pages 17–31. Springer, 2005a. 15

CS Lee and Ahmed Elgammal. Homeomorphic manifold analysis: Learning decomposable generative models for human motion analysis. *IEEE International Conference on Computer Vision*, 2005b. 16, 17

Marius Leordeanu and Martial Hebert. A spectral technique for correspondence problems using pairwise constraints. *ICCV*, 2005. 100, 112

Paul J. Lewi. Pattern recognition, reflections from a chemometric point of view. *Chemometrics and Intelligent Laboratory Systems*, 28(1):23–33, April 1995. ISSN 01697439. doi: 10.1016/0169-7439(95)80037-A. 20, 21

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014*, pages 740–755. Springer, 2014. 1, 3

David G Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial intelligence*, 31(3):355–395, 1987. 73

Juergen Luettin, NA Thacker, and SW Beet. Speaker identification by lipreading. *International Conference on Spoken Language Processing*, pages 1–4, 1996. 26

D. Marr. *Vision: A computational investigation into the human representation and processing of visual information*. W.H. Freeman, 1982. 73

I. Matthews and T.F. Cootes. Extraction of visual features for lipreading. *PAMI*, 24(2):198–213, 2002. 27, 28, 40, 41, 46, 100, 112

HARRY McGurk and JOHN MacDonald. Hearing lips and seeing voices. *Nature*, 264(23 December):746–748, 1976. 25

Liang Mei, Jingen Liu, Alfred Hero, and Silvio Savarese. Robust object pose estimation via statistical manifold modeling. *ICCV*, 2011. 50, 52, 53, 71, 74

H. Murase and S. Nayar. Visual learning and recognition of 3d objects from appearance. *International Journal of Computer Vision*, 14:5–24, 1995. 52, 77, 129

Cristianini N, Shawe-Taylor J, and Kandola JS. Spectral kernel methods for clustering. *NIPS*, 2001. 136

Ramanan Navaratnam, Andrew W Fitzgibbon, and Roberto Cipolla. The joint manifold model for semi-supervised multi-valued regression. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007. 103

Anthony O'Hagan and JFC Kingman. Curve fitting and optimal design for prediction. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–42, 1978. 21, 101

T Ojala. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *PAMI*, 24(7):971–987, 2002. 41, 112

M. Ozuysal, V. Lepetit, and P. Fua. Pose estimation for category specific multiview object localization. In *CVPR*. 2009. 64, 66, 67, 90, 94, 95

EK Patterson and S Gurbuz. Moving-talker, speaker-independent feature study, and baseline results using the CUAVE multimodal speech corpus. *EURASIP Journal on Appl. Signal Process.*, 2002(1110-8657):1189–1201, 2002. 40

Nadia Payet and Sinisa Todorovic. From contours to 3d object detection and pose estimation. In *ICCV*. 2011. 74

Bojan Pepik, Michael Stark, Peter Gehler, and Bernt Schiele. Teaching 3d geometry to deformable part models. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3362–3369. IEEE, 2012. 74

Tomaso Poggio and Federico Girosi. Network for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, 1990. 18, 30, 31, 56, 80

Gerasimos Potamianos and Chalapathy Neti. Audio-visual automatic speech recognition: An overview. *Issues in Visual and Audio-Visual Speech Processing*, 2004. 25

L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989. 28

Marc'Aurelio Ranzato, Fu-Jie Huang, Y-Lan Boureau, and Yann LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *Proc. Computer Vision and Pattern Recognition Conference (CVPR'07)*. IEEE Press, 2007. 127

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT press, 2006. 3, 10, 21, 23, 24, 101, 105, 106

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, pages 512–519. IEEE, 2014. 128

Roman Rosipal and Leonard J. Trejo. Kernel partial least squares regression in reproducing kernel hilbert space. *The Journal of Machine Learning Research*, 2:97–123, 2002. 6, 19, 21, 32, 33, 36, 55, 57, 62, 137

Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000. xii, 2

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014. URL `http://arxiv.org/abs/1409.0575`. 133

Kate Saenko and Karen Livescu. Visual speech recognition with loosely synchronized feature streams. *IEEE International Conference on Computer Vision*, 2005. 28

Mathieu Salzmann, Carl H Ek, Raquel Urtasun, and Trevor Darrell. Factorized orthogonal latent spaces. In *International Conference on Artificial Intelligence and Statistics*, pages 701–708. 2010. 103

Conrad Sanderson and Kuldip Paliwal. Identity verification using speech and face information. *Digital Signal Processing*, 14(5):449–480, sep 2004. 26

S. Savarese and F.F. Li. View synthesis for recognizing unseen poses of object classes. In *ECCV*, pages III: 602–615. 2008. 74, 90

Silvio Savarese and Li Fei-Fei. 3d generic object categorization, localization and pose estimation. *ICCV*, 2007. xv, 53, 64, 65, 66, 74, 79, 90, 95, 111

Silvio Savarese and Li Fei-fei. Multi-view Object Categorization and Pose Estimation. In *Computer Vision, SCI 285*. Springer-Verlag Berlin Heidelberg, 2010. 53

Johannes Schels, Joerg Liebelt, and Rainer Lienhart. Learning an object class representation on a continuous viewsphere. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3170–3177. IEEE, 2012. 50, 53, 71, 74

Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *CoRR*, abs/1312.6229, 2013. 124, 127

AA Shaikh, DK Kumar, and WC Yau. Lip Reading using Optical Flow and Support Vector Machines. *IEEE International Congress on Image and Signal Processing*, 1:327–330, 2010. doi: 10.1109/CISP.2010.5646264. 29

D Shiell and L Terry. Audio-Visual and Visual-Only Speech and Speaker Recognition: Issues about Theory, System Design, and Implementation. *Visual speech recognition: lip segmentation and mapping*, pages 1–38, 2009. 25, 26

Ilan Shimshoni and Jean Ponce. Finite-resolution aspect graphs of polyhedral objects. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(4):315–327, 1997. 73

Leonid Sigal and Michael J Black. Humaneva: Synchronized video and motion capture dataset for evaluation of articulated human motion. *Brown Univertsity TR*, 120, 2006. 100, 113

Josef Sivic, Bryan C. Russell, Alexei A. Efros, Andrew Zisserman, and William T. Freeman. Discovering objects and their location in images. In *ICCV*. 2005. 74

J.B. Tenenbaum and W. T. Freeman. Separating style and content with bilinear models. *Neural Computation*, 12:1247–1283, 2000. 101

Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000. 4

Damien Teney and Justus Piater. Continuous pose estimation in 2d images at instance and category levels. *2013 International Conference on Computer and Robot Vision*, 0:121–127, 2013. 66, 67, 95

Marwan Torki and Ahmed Elgammal. Regression from local features for viewpoint and pose estimation. *ICCV*, 2011. 50, 53, 66, 67, 95

Pavan Turaga, Ashok Veeraraghavan, Anuj Srivastava, and Rama Chellappa. Statistical computations on grassmann and stiefel manifolds for image and video-based recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(11):2273–2286, 2011. 88

Raquel Urtasun and Trevor Darrell. Discriminative gaussian process latent variable model for classification. In *Proceedings of the 24th international conference on Machine learning*, pages 927–934. ACM, 2007. 117

Raquel Urtasun, David J. Fleet, and Pascal Fua. 3D people tracking with gaussian process dynamical models. *CVPR*, pages 238–245, 2006. 11, 102

Raquel Urtasun, David J Fleet, Andreas Geiger, Jovan Popović, Trevor J Darrell, and Neil D Lawrence. Topologically-constrained latent variable models. In *Proceedings of the 25th international conference on Machine learning*, pages 1080–1087. ACM, 2008. 114

M. Alex O. Vasilescu and Demetri Terzopoulos. Multilinear subspace analysis of image ensembles. In *CVPR*. 2003. 101

Grace Wahba. Support vector machines, reproducing kernel Hilbert spaces and the randomized GACV. *Advances in Kernel Methods-Support Vector Learning*, 984, 1999. 21, 101

Jack Wang, Aaron Hertzmann, and David M Blei. Gaussian process dynamical models. In *Advances in neural information processing systems*, pages 1441–1448. 2005. xv, 11, 99, 101, 102, 119

Jack M Wang, David J Fleet, and Aaron Hertzmann. Multifactor gaussian process models for style-content separation. In *Proceedings of the 24th international conference on Machine learning*, pages 975–982. ACM, 2007. x, xv, 102, 107, 118, 119, 120

Jutta Willamowski, Damian Arregui, Gabriella Csurka, Christopher R. Dance, and Lixin Fan. Categorizing nine visual classes using local appearance descriptors. In *IWLAVS*. 2004. 74

H. Wold. Soft Modeling by Latent Variables; the Nonlinear Iterative Partial Least Squares Approach. *Perspectives in Probability and Statistics. Papers in Honour of M. S. Bartlett*, pages 520 – 540, 1975. 19, 20, 21

Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2014. xi, 131, 143, 144, 145

J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? *ArXiv e-prints*, November 2014. 124, 127

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013. URL `http://arxiv.org/abs/1311.2901`. 124, 128, 129

Haopeng Zhang, Tarek El-Gaaly, Ahmed Elgammal, and Zhiguo Jiang. Joint object and pose recognition using homeomorphic manifold analysis. In *AAAI*. 2013. x, xi, 8, 15, 50, 53, 64, 65, 68, 70, 71, 75, 76, 79, 93, 94, 95, 129, 143, 145

Guoying Zhao. Lipreading with local spatiotemporal descriptors. *IEEE Transactions on Multimedia*, pages 1–11, 2009. xiii, 27, 28, 29, 40, 41, 42, 45, 46

Ziheng Zhou, Guoying Zhao, and M Pietikainen. Towards a practical lipreading system. *Computer Vision and Pattern Recognition*, 2011. 28, 45, 46