Kernel Methods for Unsupervised Domain Adaptation

by

Boqing Gong

A Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(COMPUTER SCIENCE)

August 2015

Acknowledgements

This thesis concludes a wonderful four-year journey at USC. I would like to take the chance to express my sincere gratitude to my amazing mentors and friends during my Ph.D. training.

First and foremost I would like to thank my adviser, Prof. **Fei Sha**, without whom there would be no single page of this thesis. Fei is smart, knowledgeable, and inspiring. Being truly fortunate, I got an enormous amount of guidance and support from him, financially, academically, and emotionally. He consistently and persuasively conveyed the spirit of adventure in research and academia of which I appreciate very much and from which my interests in trying out the faculty life start. On one hand, Fei is tough and sets a high standard on my research at "home"—the TEDS lab he leads. On the other hand, Fei is enthusiastically supportive when I reach out to conferences and the job market. These combined make a wonderful mix. I cherish every mind-blowing discussion with him, which sometimes lasted for hours.

I would like to thank our long-term collaborator, Prof. **Kristen Grauman**, whom I see as my other academic adviser. Like Fei, she has set such a great model for me to follow on the road of becoming a good researcher. She is a deep thinker, a fantastic writer, and a hardworking professor. I will never forget how she praised our good work, how she hesitated on my poor proposals, that she stayed up with us before NIPS deadlines, how she helped me refine the slides of my first oral presentation at CVPR, and many others. Thank her for hosting me during the summer 2012.

I give special thanks to Prof. **Trevor Darrell**, who pioneered the research topic of this thesis. Besides, I respect him also for how he wholeheartedly inspires and promotes young researchers like me and for his help throughout my job search.

I thank Prof.s **Jianzhuang Liu**, **Xiaogang Wang**, and **Xiaoou Tang**, who prepared me for the Ph.D. program when I pursued the M.Phil. degree at CUHK. Without them the thesis would have been delayed for at least one year. I particularly thank Xiaogang for helping me with the job search.

I would like to thank Prof. **Gaurav S. Sukhatme** and Prof. **Shrikanth S. Narayanan** for graciously serving in my thesis committee. Especially Prof. Sukhatme set up a mock interview for me before I went out for the job interviews and followed up to give suggestions afterwards. I have gained a lot from his perspective.

Behind this thesis, my Ph.D. life has been very enjoyable thanks to my gifted and fun labmates. **Meihong** and **Dingchao** brought me many precious memories at my early stage at USC. **Tomer**, **Erica**, **Anand**, and **Franziska** opened another door for me to learn about different cultures. Thank **Zhiyun**, **Kuan**, **Ali**, and **Ke** for both professional discussions and random chats. I give special thanks to **Harry** and **Beer** for their hardworking while we collaborated on the computer vision projects. Harry is always energetic and passionate and Beer often stays up crazily late. The same appreciation goes to **Wenzhe**, **Dong**, and **Yuan**. It is great to have so many gifted

colleagues working on different projects, talking about diverse topics, and complementing each other from a wide range of aspects.

My family is the most basic source of my life energy. This thesis is dedicated to my family. My parents are incredible lifelong role models who work hard, play harder, and provide me and my younger sister, Liya, the perfect blend of freedom and guidance. It is priceless growing up with my younger sister. We shared and fighted for candies and together we fooled our parents. Mom, Dad, and Liya, thank you for your endless support and your unconditional confidence in me. Finally, I arrive at the hardest part, thanking my beloved wife and fabulous company, Zhaojun, though words cannot faithfully express my feelings. She teaches me love, makes me happy, cheers with me, argues with me, and has been the best gift life gives me. Every moment thinking of or looking at her, I cannot help smiling. Thank you Zhaojun, for making me the luckiest man.

Table of Contents

List of Tables

List of Figures

Abstract

In many applications (computer vision, natural language processing, speech recognition, *etc.*), the curse of domain mismatch arises when the test data (of a target domain) and the training data (of some source domain(s)) come from different distributions. Thus, developing techniques for domain adaptation, *i.e.*, generalizing models from the sources to the target, has been a pressing need. When the learner has access to only unlabeled data from the target domain (and labeled data from the source domain), the problem is called *unsupervised domain adaptation*. Advances in domain adaptation can significantly increase our capability to deploy autonomous and intelligent systems in challenging environments where uncertainty prevails.

This thesis work provides a comprehensive set of techniques with multiple levels of focus on unsupervised domain adaptation, including learning domain-invariant feature representations in order to eliminate the discrepancy between the source domain and the target domain, manipulating data instances to match the distributions of two domains, quantizing the "adaptabilities" of different source domains given a particular target domain, and discovering latent domains from heterogeneous data so the individual domains can be better and more efficiently modeled. We demonstrate the effectiveness of the developed methods on well-benchmarked datasets and tasks (visual object recognition, sentiment analysis, and cross-view human activity recognition).

In terms of domain adaptation algorithms, this thesis develops two complementary approaches using kernel methods, one to infer domain-invariant geodesic flow kernels (GFKs) and the other to directly match the underlying distributions of two domains. GFK models data by subspaces and interpolates an infinite number of phantom domains between the source domain and the target domain. We then use the "kernel trick'' to average out domain-specific idiosyncrasies and arrive at a domain-invariant kernel. Built upon GFK, we propose an approach to identifying the most adaptable data instances of the source domain, named as landmarks, to the target domain. Due to that the landmarks are more similar to the target domain in the sense of their underlying distributions, adapting from the landmarks gives rise to better performance on the target than adapting from the original source domain.

This thesis also contributes to other aspects of domain adaptation. We make some preliminary efforts on answering the open question of how to evaluate the "adaptability" of a source domain to the target domain. This results in a rank-of-domains (ROD) metric, which exploits both geometrical and statistical discrepancies between two domains. Besides, this thesis raises the concern about how to define or represent a domain with real data. While by a domain we refer to the underlying distribution of the observed data, the distribution is often unknown. A standard practice has been equating datasets with domains. However, our studies show that this is not necessarily the best for the adaptation tasks. An automatic algorithm is instead proposed to "reshape" the data into domains which are better in terms of the adaptation performance to the target domain.

To further explore kernels, which play the central role in our approaches to domain adaptation, this thesis concludes by researching kernels in a probabilistic model, determinantal point process (DPP). We propose a novel model, sequential DPP, for supervised video summarization and derive a large-margin training algorithm for learning the kernels in DPPs.

Part I

Background

Chapter 1

Introduction

We are at the data-centric era. There has been a rapid growth of data from various areas during the recent years. Though it is difficult to have a complete understanding and characterization of the "big data" phenomenon, we can gain some insights from some representative exemplars.

- Internet based companies and services have boosted the scale of publicly accessible datasets out of most people's imagination. There are hundreds of millions of users of Facebook, Twitter, Youtube, *etc.* They (we) had contributed to the 300 petabytes (300×10^{15} bytes) user data Facebook hosted as of Nov. 6th, 2013[1], have been posting more than 500 million Tweets daily[2], and have been uploading more than 432,000 hours of videos to Youtube every day[3].

- Research in both science and engineering is also generating a huge volume of observational and/or synthesized data. It becomes necessary to collect and analyze large-scaled datasets in order to make new discovery, to extract new knowledge, or to build more robust systems. For instance, the recent impressive progress on visual object recognition from images and videos is largely due to the millions of human labeled images (Deng et al., 2009) (and deep neural networks with millions of parameters (Krizhevsky et al., 2012)).

- The "big data" also emerge from the domain of public security as a result of monitoring and surveillance. Take London for example. About 422,000 CCTV cameras perching around London survey happenings in the city 24/7.

There is accordingly a pressing need for scalable technologies of data management (storage, retrieval, indexing, *etc.*) and data analysis. This thesis focuses on the latter, of which machine learning plays the central role. Data analysis aims to infer useful knowledge and to make new discovery from data. Thanks to the decades of research and industrial development on data-centric algorithms, theories, and systems, it is reasonable to foresee many promising and exciting opportunities revolving around data analysis.

Meanwhile, however, the work on data analysis is encountering many grand new challenges along with the growing massive data collections. How to build models and deal with highly

[1]For more details please see `https://www.facebook.com/notes/facebook-engineering/presto-interacting-with-petabytes-of-data-at-facebook/10151786197628920`.

[2]Statistics as of May 20, 2015 from `https://about.twitter.com/company`.

[3]Statistics as of May 20, 2015 from `https://www.youtube.com/yt/press/statistics.html`.

distributed data? It involves a mix of several aspects of the machine learning algorithms to answer this question, such as distributed optimization, communication costs between different machines, and so on. How to learn from streaming data? How shall we trade-off the optimization costs and the generalization capabilities of machine learning algorithms?

Another major challenge is the sampling bias problem. Conventional statistical machine learning revolves on an (overly) simplified assumption that the training data, from which the algorithms learn, are drawn *i.i.d.* from the same distribution as the test data, to which the learned models are applied. This assumption and the corresponding algorithms are fundamentally restrictive, being frequently challenged in numerous situations. For instance, a pedestrian detection and avoidance computer vision system on automobiles faces very different data when weather patterns change, when cameras age, or simply when people drive to new locations. In other words, the training and test data are often *mismatched*.

Sampling bias is particularly severe and prevalent in modern massive data collections, whose sub-collections often follow disparate sampling criteria. Indeed, a large collection of data may consist of sub-collections by different research groups and companies from different collection sources whose characteristics vary over time. These exemplify the limitations of conventional statistical learning algorithms—many empirical studies have shown that when the statistical properties change from the training to the test data, the performance of the learning systems often degrades significantly (Torralba & Efros, 2011; Dollár et al., 2009; Daumé III & Marcu, 2006; Blitzer et al., 2006).

As a result, practical autonomous systems inevitably suffer from the sampling bias. The systems are often deployed to new target environments. It is unrealistic to attempt to reproduce all sorts of the target environment when one develops the systems, not to mention that real-life environments are often not lab-reproducible. Besides, the systems naturally degrade and the outside environment changes over time. In sum, there is no way to completely avoid the mismatch. Instead, it is highly desirable to have a new statistical machine learning paradigm to explicitly deal with the mismatches in data.

This thesis concentrates on addressing this challenge in the framework of unsupervised domain adaptation.

1.1 Domain adaptation

Imagine that we are to deploy an Android application to recognize objects in images captured with mobile phone cameras. Instead of demanding that users provide labels to our learning algorithms, can we train classifiers with existing tagged images or labeled vision datasets, such as Flickr photos, LabelMe (Russell et al., 2008), or ImageNet (Deng et al., 2009), and hope the classifiers will still work well on mobile camera test images? See Fig. 1.1 for an illustration.

Our intuition says no. We suspect that the strong distinction between images in those datasets and typical mobile phone images will cripple those classifiers. Indeed, a stream of studies have shown that when image classifiers are evaluated outside of their training datasets, the performance degrades significantly (Torralba & Efros, 2011; Dollár et al., 2009; Perronnin et al., 2010). The culprit is clear: the visual appearance of even the same object varies significantly across different datasets as a result of many factors, including imaging devices, photographers' preferences, background, and illumination. These idiosyncrasies often cause a substantial mismatch between the training and the testing data distributions.

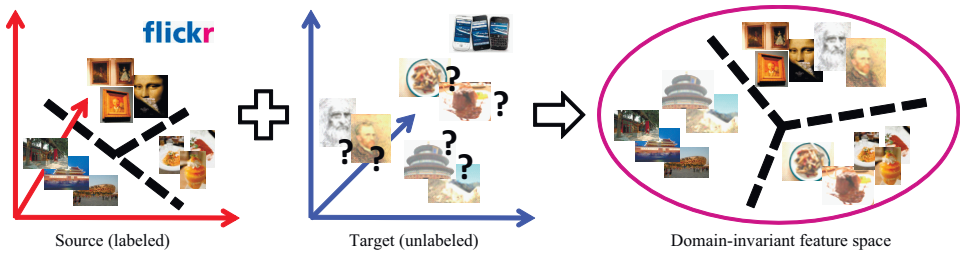

Figure 1.1: Schematic illustration of unsupervised domain adaptation. The objective is to classify unlabeled data from the target domain which has different characteristics from the source domain where annotated data are provided. The central idea behind the approaches proposed in this thesis is to use the labeled data to build a classifier in a domain-invariant feature space learned using kernel methods. We use data from both domains to infer the feature space such that the adapted classifier from the source domain will work well on the target domain.

Beyond image recognition, mismatched training and testing conditions are also abundant in other computer vision tasks (Duan et al., 2009a; Wang & Wang, 2011; Jain & Learned-Miller, 2011; Duan et al., 2010), speech recognition (Leggetter & Woodland, 1995b; Reynolds et al., 2000; Huang & Hasegawa-Johnson, 2008) and text classification and analysis (Blitzer et al., 2007, 2006; Jiang & Zhai, 2007; Glorot et al., 2011).

In all these pattern recognition tasks, there is a common theme. There are two distinct types of datasets: one from a **source** domain and the other from a **target** domain. The source domain contains a large amount of labeled data such that a classifier can be reliably built. The target domain refers broadly to a related dataset that has different characteristics compared to the source domain. Since conventional learning algorithms rely heavily on the assumption that data used for training and testing are drawn from the same distribution, they are inadequate in the face of such mismatched domains. Thus, the main objective is to *adapt* classifiers trained on the source domain to the target domain to attain good performance there[4].

How can we build classifiers and other statistical machine learning models that are robust to mismatched distributions? Techniques for addressing this challenge have been investigated under the names of domain adaptation, covariate shift, or transfer learning (Shimodaira, 2000; Daumé III & Marcu, 2006; Pan & Yang, 2010; Gretton et al., 2009). When there are no labeled data from the target domain to help learning, the problem is called **unsupervised domain adaptation** (Blitzer et al., 2007, 2006; Gopalan et al., 2011; Gong et al., 2012b; Chen et al., 2011a). In contrast, when some labeled data from the target domain are accessible, the problem is similar to semi-supervised learning and is referred to as **semi-supervised domain adaptation** (Daumé III et al., 2010; Bergamo & Torresani, 2010; Saenko et al., 2010). In either case, however, the labeled target data alone are insufficient to construct a good classifier. Thus, how to effectively leverage *unlabeled target data* is key to domain adaptation.

[4]Note that we assume the sets of possible labels are the same across domains.

1.2 Contributions

This thesis work provides a comprehensive set of techniques with multiple levels of focus on unsupervised domain adaptation, including learning domain-invariant feature representations in order to eliminate the discrepancy between the source domain and the target domain (Chapter 4), manipulating data instances to match the distributions of two domains (Chapter 5), discovering latent domains from heterogeneous data so the individual domains can be better and more efficiently modeled (Chapter 7), and quantizing the "adaptabilities" of different source domains given a particular target domain (Chapter 6).

In terms of the methodologies behind the thesis work, we use kernel methods to develop our solutions to the research questions around domain adaptation. One of our approaches gives rise to a new kernel function which is invariant to the mismatched domains, while the others take advantage of the kernel embedding of probabilistic distributions. We shall provide a gentle tutorial to the kernel methods in Chapter 3. Moreover, in Chapter 8 we explore more applications of the kernels in probabilistic models for which we derive a new large-margin learning algorithm.

1.3 Thesis outline

The remaining of this thesis is outlined as follows.

Chapter 2 is a survey on unsupervised and semi-supervised domain adaptation. We present the existing work in the literature by organizing them to three broad categories, re-weighting data instances, learning feature representations, and adapting background or source models.

Chapter 3 gives a gentle tutorial to kernel methods. We particularly describe the "kernel trick" and the kernel embedding of distributions in detail.

Chapter 4 presents our geodesic flow kernel (GFK) to infer new feature (kernel) representations that are resilient to the mismatches between two domains.

Chapter 5 builds upon GFK and introduces a novel inherent structure in domain adaptation, landmarks. The key insight is that not all instances from the source domain are created equally in terms of adaptability to the target domain. By automatically identifying and leveraging the landmarks we will have an easier domain adaptation task than the original one which involves the whole source domain.

Chapter 6 defines the rank-of-domains (ROD) metric used to measure the adaptabilities of different source domains to the target domain.

Chapter 7 examines the notation of *domains*. We show that it is essentially hard to manually group visual data (images, videos, *etc.*) to represent the underlying domains. Instead, we develop a nonparametric method to automatically reshape given datasets to domains.

Chapter 8 exploits kernel methods in the probabilistic model, determinantal point process (DPP). We develop a large-margin learning objective and derive a new DPP model to handle the sequential structure in video data.

Chapter 9 demonstrates the effectiveness of the developed methods on well benchmarked datasets and tasks (visual object recognition, sentiment analysis, and cross-view human activity recognition).

We conclude this thesis and remark on the future directions by Chapter 10.

1.4 Previously published work

Chapter 4 and Chapter 6 draw significantly from our work published in CVPR 2012 (Gong et al., 2012b) and IJCV 2014 (Gong et al., 2014b). Chapter 5 is mainly based on our paper of ICML 2013 (Gong et al., 2013b). Chapter 7 corresponds to our NIPS 2013 paper (Gong et al., 2013a). Chapter 8 draws from our NIPS 2014 (Gong et al., 2014a) and UAI 2015 (Chao et al., 2015). Some of the results in this thesis have also been presented at the NIPS 2012 Workshop of Large Scale Visual Recognition and Retrieval (Gong et al., 2012a) and the ECCV 2014 Workshop of Transferring and Adapting Source Knowledge (TASK) in Computer Vision.

Chapter 2

Domain Adaptation: A Survey

Domain adaptation has been extensively studied in many areas, including statistics and machine learning (Shimodaira, 2000; Huang et al., 2006; Ben-David et al., 2007; Pan & Yang, 2010), speech and language processing (Daumé III, 2007; Blitzer et al., 2006; Leggetter & Woodland, 1995b), and more recently computer vision (Bergamo & Torresani, 2010; Gopalan et al., 2011; Saenko et al., 2010; Kulis et al., 2011). In this chapter, we provide a survey on domain adaptation and focus on the methodologies developed in the literature, regardless of their specific application domains. Among them, of particular relevance to our work are those learning domain-invariant feature representations and thus enabling effortlessly transferring statistical machine learning models from the source domain to the target domain (Ben-David et al., 2007; Blitzer et al., 2006, 2007; Daumé III, 2007; Pan et al., 2009).

We begin with a widely used assumption in domain adaptation called covariate shift. It was first proposed in the seminal work (Shimodaira, 2000) in the context of learning predictive densities. Through examining the bias and variance of the maximum likelihood estimators (MLEs), Shimodaira derived a direct solution to covariate shift, instance re-weighting. Most early works on domain adaptation center around how to effectively estimate the instance weights (Sugiyama & Müller, 2005; Huang et al., 2006; Sugiyama et al., 2008; Yamada et al., 2011). In Section 2.1, we describe the covariate shift assumption under empirical risk estimation, a broader context than MLE, and review the corresponding instance re-weighting approaches.

While instance re-weighting directly follows the statistical analysis, it does not offer the modeling flexibility of incorporating the domain knowledge of data representations. Some features of the data are generic across different domains and hence can significantly ease the adaptation if we train machine learning models upon them. Take sentiment analysis of user reviews of different products for instance. Some very predictive words like "highly recommend" and "not worth it" show up in the reviews of various products. We review the feature learning based approaches to domain adaptation in Section 2.2.

Additionally, in Section 2.3 we visit some methods which adapt either a "background" model or the model trained on the source domain to the target domain. In probabilistic modeling, this can be straightforwardly implemented by taking the models on the source domain as the prior of the models on the target domain. Besides, it is also interesting to draw some analogy between domain adaptation and multi-task learning, regarding the domains as different tasks. Finally, domain knowledge acts as extra cues to facilitate adapting source classifiers to the target in some specific applications.

2.1 Covariance shift and instance weighting

In statistics, a covariate is a variable which is possibly predictive of the outcome under study. It is also termed as explanatory variable, independent variable, or predictor in some scenarios. *Covariate shift* refers to the situation that the marginal distribution $P_{\mathcal{S}}(\boldsymbol{x})$ of the source domain $\mathcal{S}$ differs from $P_{\mathcal{T}}(\boldsymbol{x})$, the marginal distribution of the target domain $\mathcal{T}$ for evaluation, while the conditional distribution $P(y|\boldsymbol{x}) = P_{\mathcal{S}}(y|\boldsymbol{x}) = P_{\mathcal{T}}(y|\boldsymbol{x})$ is shared between the two domains. This is perhaps the most widely used assumption in domain adaptation, either implicitly or explicitly. Extensive empirical studies show that even if the assumption does not strictly hold, the algorithms proposed using the assumption still perform reasonably well on the target domain.

As a special form of domain adaptation, covariate shift was first studied in (Shimodaira, 2000). Before that, it was used to detect model mis-specification (White, 1981) of nonlinear regression models. It also has some overlap with sample selection bias (Heckman, 1979; Vella, 1998), which mainly refers to non-random selections of individuals, groups, or data from the population intended to be analyzed.

Next we apply the covariate shift assumption to the principle of empirical risk minimization and arrive at an instance re-weighting scheme. Similar derivations can be found in (Shimodaira, 2000) and (Huang et al., 2006).

2.1.1 Covariate shift in empirical risk minimization

Consider supervised learning problems via empirical risk minimization (ERM). We have an input space $\mathcal{X}$ and an output space $\mathcal{Y}$. The learner in general aims to find an optimal decision function $h^\star : \mathcal{X} \mapsto \mathcal{Y}$ from a hypothesis set $\mathcal{H}$, so as to minimize the generalization error (also called the expected risk),

$$R(h; L, P) = \mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[L(h(\boldsymbol{x}), y)\right], \quad h \in \mathcal{H} \tag{2.1}$$

where $L : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}$ is a loss function and $P(\boldsymbol{x}, y)$ is a joint distribution of $\boldsymbol{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$.

Note that the distribution $P(\boldsymbol{x}, y)$ is often unknown to the learner. Instead, the learner receives a training sample $S = \{(\boldsymbol{x}_m, y_m)\}_{m=1}^{\mathsf{M}}$ drawn *i.i.d.* according to $P(\boldsymbol{x}, y)$, and can hence approximate $R(h; L, P)$ using the empirical risk,

$$R_{\text{emp}}(h; L, S) = \frac{1}{\mathsf{M}} \sum_{m=1}^{\mathsf{M}} L(h(\boldsymbol{x}_m), y_m), \tag{2.2}$$

which is an unbiased estimator of the expected risk $R(h; L, P)$. The learner then returns a hypothesis $\hat{h} \in \mathcal{H}$ according to the ERM principle, *i.e.*,

$$\hat{h} \leftarrow \arg\min_{h\in\mathcal{H}} R_{\text{emp}}(h; L, S) + \lambda\Omega(h),$$

where $\Omega(h)$ is a regularizer and the (hyper-)parameter $\lambda > 0$ balances the strengths of the empirical risk and the regularization term.

The ERM hypothesis $\hat{h}$ achieves the minimal risk on the observed training sample S, while the most of interest is the true risk under the distribution P. Fortunately, the "goodness" of $\hat{h}$

can be bounded by the sample size M and the complexity or capacity of the hypothesis set $\mathcal{H}$ measured in various forms (Vapnik, 2000; Mohri et al., 2012).

The ERM principle becomes more involved in domain adaptation. Denote by $\mathcal{S}$ and $\mathcal{T}$ the source and the target domains, respectively, associated with joint distributions $P_{\mathcal{S}}(\boldsymbol{x}, y)$ and $P_{\mathcal{T}}(\boldsymbol{x}, y)$ over $\mathcal{X} \times \mathcal{Y}$. The learner receives a labeled sample $S = \{(\boldsymbol{x}_m, y_m)\}_{m=1}^{\mathsf{M}}$ drawn *i.i.d.* from the source domain according to the source distribution $P_{\mathcal{S}}(\boldsymbol{x}, y)$, and an *i.i.d.* sample $T = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^{\mathsf{N}}$ from the target domain according to the target distribution $P_{\mathcal{T}}(\boldsymbol{x}, y)$. The sample T of the target domain is either too small to give rise to a reliable approximation of the expected risk, or some or even all of its labels $\{y_n\}_{n=1}^{\mathsf{N}}$ are not revealed to the learner.

Motivated by importance sampling, we can harvest the source sample S to estimate the risk of the target,

$$
\begin{aligned}
R(h; L, P_{\mathcal{T}}) &= \mathbb{E}_{(\boldsymbol{x},y)\sim P_{\mathcal{T}}} \left[L(h(\boldsymbol{x}), y) \right] && (2.3)\\
&= \mathbb{E}_{(\boldsymbol{x},y)\sim P_{\mathcal{S}}} \left[\frac{P_{\mathcal{T}}(\boldsymbol{x}, y)}{P_{\mathcal{S}}(\boldsymbol{x}, y)} L(h(\boldsymbol{x}), y) \right] && (2.4)\\
&= R(h; L_\omega, P_{\mathcal{S}}), && (2.5)
\end{aligned}
$$

where we arrive at the expected risk on the source domain but with a weighted loss function $L_\omega(\boldsymbol{x}, y) = \omega(\boldsymbol{x}, y) L(\boldsymbol{x}, y)$. The weight is $\omega(\boldsymbol{x}, y) = \frac{P_{\mathcal{T}}(\boldsymbol{x},y)}{P_{\mathcal{S}}(\boldsymbol{x},y)} = \frac{P_{\mathcal{T}}(\boldsymbol{x})}{P_{\mathcal{S}}(\boldsymbol{x})}$ under covariate shift, so in what follows we denote by $\omega(\boldsymbol{x})$ the weighting function $\omega(\boldsymbol{x}, y)$. The existence of the instance weight $\omega(\boldsymbol{x})$ depends on the following conditions.

Assumption 1 $P_{\mathcal{T}}(\boldsymbol{x})$ has the support within that of $P_{\mathcal{S}}(\boldsymbol{x})$.

Assumption 2 $P_{\mathcal{S}}(\boldsymbol{x})$ is strictly positive over the support of $P_{\mathcal{T}}(\boldsymbol{x})$. Note that since one has to use the source sample $S = \{(\boldsymbol{x}_m, y_m)\}_{m=1}^{\mathsf{M}}$ to approximate the expected risk $R(h; L_\omega, P_{\mathcal{S}})$ and $P_{\mathcal{S}}(\boldsymbol{x}_m, y_m) > 0$ for all $(\boldsymbol{x}_m, y_m) \in S$, this assumption is practically mild.

In practice, often neither $P_{\mathcal{S}}(\boldsymbol{x})$ nor $P_{\mathcal{T}}(\boldsymbol{x})$ is known by the learner. As a result, one has to estimate the instance weights before calculating the empirical risk of the target using the source sample (*cf.* eq. (2.5)). To this end, Shimodaira discussed estimating $P_{\mathcal{S}}(\boldsymbol{x})$ and $P_{\mathcal{T}}(\boldsymbol{x})$ using parametric or nonparametric methods (Shimodaira, 2000). However, density estimation easily suffers from the curse of dimensionality. Next, we review some methods on directly learning the weights, without the intermediate step of estimating the source or the target marginal distributions.

2.1.2 Directly estimating instance weights

Observing that $\omega(\boldsymbol{x}) P_{\mathcal{S}}(\boldsymbol{x}) = P_{\mathcal{T}}(\boldsymbol{x})$, we can hence learn a weighting function $\beta(\boldsymbol{x}) \approx \omega(\boldsymbol{x})$ from the data, such that it minimizes the divergence between $\beta(\boldsymbol{x}) P_{\mathcal{S}}(\boldsymbol{x})$ and $P_{\mathcal{T}}(\boldsymbol{x})$, or between $\beta(\boldsymbol{x})$ and $P_{\mathcal{T}}(\boldsymbol{x}) / P_{\mathcal{S}}(\boldsymbol{x})$, under some constraints,

$$
\min_{\beta(\boldsymbol{x})} D(\beta(\boldsymbol{x}) P_{\mathcal{S}}(\boldsymbol{x}), P_{\mathcal{T}}(\boldsymbol{x})), \quad \text{s.t.}, \beta(\boldsymbol{x}) \geq 0, \ \mathbb{E}_{\boldsymbol{x}\sim P_{\mathcal{S}}(\boldsymbol{x})} \beta(\boldsymbol{x}) = 1 \tag{2.6}
$$

where $D(\cdot, \cdot)$ is some discrepancy measure like KL-divergence. Obviously $\omega(\boldsymbol{x})$ is among the best solutions $\{\beta^\star(\boldsymbol{x})\}$ to eq. (2.6). If $D(\cdot, \cdot)$ is a metric and is (strictly) convex (*e.g.*, the maximum mean discrepancy shown below), we will have a unique solution $\beta^\star(\boldsymbol{x}) = \omega(\boldsymbol{x})$. Therefore,

the quality of the directly estimated instance weights depend on which discrepancy measure to use and how to deal with the unknown distributions $P_{\mathcal{S}}(\boldsymbol{x})$ and $P_{\mathcal{T}}(\boldsymbol{x})$ by the samples S and T, respectively.

Maximum mean discrepancy One of the most popular approaches to directly estimating $\omega(\boldsymbol{x})$ is by the maximum mean discrepancy (MMD) (Huang et al., 2006; Gretton et al., 2009). The main idea is to define the MMD measure between two distributions in a Reproducing Kernel Hilbert Space (RKHS). We provide the details of MMD here considering that we shall use it in our approaches in the following chapters.

Abuse the notation $\mathcal{H}$ a little bit and use it to denote an RKHS associated with a characteristic kernel function $k(\cdot,\cdot)$ (*e.g.*, Gaussian RBF) (Sriperumbudur, 2010). The following results hold.

- The embedding $\mu[P_{\mathcal{S}}(\boldsymbol{x})]$ is injective,

$$\mu[P_{\mathcal{S}}(\boldsymbol{x})] \triangleq \mathbb{E}_{\boldsymbol{x}\sim P_{\mathcal{S}}}[k(\boldsymbol{x},\cdot)], \quad \mu_{\text{emp}}[X_{\mathcal{S}}] \triangleq \frac{1}{\mathsf{M}}\sum_{m=1}^{\mathsf{M}} k(\boldsymbol{x}_m,\cdot) \tag{2.7}$$

  where $\mu_{\text{emp}}[X_{\mathcal{S}}]$ is correspondingly the empirical embedding of $\mu[P_{\mathcal{S}}(\boldsymbol{x})]$. Here we use the source marginal distribution $P_{\mathcal{S}}(\boldsymbol{x})$ for illustration, but please beware that the conclusion holds for all distributions.

- The so called maximum mean discrepancy (MMD)[1]

$$\text{MMD}(P_{\mathcal{S}}(\boldsymbol{x}), P_{\mathcal{T}}(\boldsymbol{x})) = \|\mu[P_{\mathcal{S}}(\boldsymbol{x})] - \mu[P_{\mathcal{T}}(\boldsymbol{x})]\|_{\mathcal{H}} \tag{2.8}$$

$$\text{MMD}_{\text{emp}}(X_{\mathcal{S}}, X_{\mathcal{T}}) = \left\| \frac{1}{\mathsf{M}}\sum_{m=1}^{\mathsf{M}} k(\boldsymbol{x}_m,\cdot) - \frac{1}{\mathsf{N}}\sum_{n=1}^{\mathsf{N}} k(\boldsymbol{x}_n,\cdot) \right\|_{\mathcal{H}} \tag{2.9}$$

  is a metric, implying that $\text{MMD}(P_{\mathcal{S}}, P_{\mathcal{T}}) = 0 \Leftrightarrow P_{\mathcal{S}} = P_{\mathcal{T}}$.

Huang *et al.* used the (empirical) MMD for the distribution discrepancy in eq. (2.6) to learn the instance weights (Huang et al., 2006). The resulting problem is a quadratic programming. Similar to other kernel methods which have the nice property of handling nonlinearity in linear forms, MMD offers a "linear" method to compare the high-order statistics of two distributions. In domain adaptation, it has been used not only to learn the instance weights (Huang et al., 2006; Zhang et al., 2013; Sun et al., 2011), but also to develop transfer component analysis (Pan et al., 2009), transfer multiple kernel learning (Duan et al., 2012a), discovering landmarks (Gong et al., 2013b) and latent domains (Gong et al., 2013a), *etc.*

Despite of the popularity of MMD, it is notoriously hard to choose the optimal kernel(s) for MMD (*e.g.*, how to set the bandwidth in Gaussian RBF) in domain adaptation, considering that there are not sufficient labeled data in the target domain for cross-validation. There exist some attempts to tackling this issue (Cortes et al., 2008; Gretton et al., 2012; Sriperumbudur et al., 2009; Iyer et al., 2014; Yu & Szepesvári, 2012). However, they are often limited to specific application scenarios. Further, the theory analyses in these works are, unfortunately, not refined

[1]The name of maximum mean discrepancy was termed by (Gretton et al., 2006), probably because MMD can be derived from the integral probability metric (Müller, 1997) which involves a supremum operation.

enough; the bounds are unrelated with the choice of the kernel as long as the kernel is characteristic or universal (Steinwart, 2002b). As an alternative to selecting the optimal kernel, we use a cohort of Gaussian RBF kernels to compare two domains at different scales. The solution to the domain adaptation problem combines the output obtained from all the scales. This approach introduces extra computation costs in the exchange of avoiding choosing a single kernel. Details are presented in Chapter 5.

Parameterizing the instance weight $\omega(\boldsymbol{x})$ Sugiyama *et al.* used the Kullback-Leibler divergence for the discrepancy between $P_{\mathcal{T}}(\boldsymbol{x})$ and $\beta(\boldsymbol{x})P_{\mathcal{S}}(\boldsymbol{x})$ in eq. (2.6) (Sugiyama et al., 2008). Additionally, the authors also parameterized the instance weighting function $\beta(\boldsymbol{x})$ by a linear combination of some non-negative basis functions,

$$\beta(\boldsymbol{x}) = \sum_{l=1}^{\mathsf{b}} \alpha_l \psi_l(\boldsymbol{x}), \quad \psi_l(\boldsymbol{x}) = \text{RBF}(\boldsymbol{x}, \boldsymbol{x}_{n_l}), \quad \boldsymbol{x}_{n_l} \in \mathcal{T}, \tag{2.10}$$

where each basis function $\psi_l(\boldsymbol{x})$ is pre-defined as a Gaussian RBF kernel centered around a data point randomly selected from the target domain. The authors name this method as Kullback-Leibler Importance Estimation Procedure (KLIEP).

Thanks to this form of parameterization, KLIEP reduces the number of unknowns from M, the number of training instances of the source domain, in MMD to b, the number of basis functions. Besides, the weighting function $\beta(\boldsymbol{x})$ can be extended out of the training sample. One benefit of the out-of-sample extension is that one can use cross-validation to tune the (hyper-)parameters — we can compute the instance weights of the validation set by eq. (2.10). In contrast, it is unclear how to apply cross-validation to MMD.

This form of parameterizing $\beta(\boldsymbol{x})$ is also used in (Kanamori et al., 2009a,b), where the discrepancy is specified as the squared errors between $\beta(\boldsymbol{x})$ and $P_{\mathcal{T}}(\boldsymbol{x})/P_{\mathcal{S}}(\boldsymbol{x})$. The overall formulation, called LSIF, is a quadratic form. It is computationally more efficient than KLIEP.

LSIF is carefully analyzed and extended in several aspects. In (Kanamori et al., 2009a,b), the authors also developed an unconstrained version called uLSIF, which comes with a closed-form solution. Later, uLSIF is extended to rectify the pitfall that $\omega(\boldsymbol{x}) = P_{\mathcal{T}}(\boldsymbol{x})/P_{\mathcal{S}}(\boldsymbol{x})$ could be arbitrarily large at some $\boldsymbol{x}$ where $P_{\mathcal{S}}(\boldsymbol{x})$ is small (Yamada et al., 2011). The solution is to replace $P_{\mathcal{S}}(\boldsymbol{x})$ by $P_{\mathcal{S}}^{\gamma} = (1-\gamma)P_{\mathcal{S}}(\boldsymbol{x}) + \gamma P_{\mathcal{T}}(\boldsymbol{x})$, $0 < \gamma < 1$, such that the new instance weighting function is always upper-bounded by γ^{-1}.

2.1.3 Modeling assumptions analogous to covariate shift

There are some other modeling assumptions in domain adaptation which share the same spirit as covariate shift and also lead to natural solutions by re-weighting the source instances. We briefly discuss two of them, sample selection bias and target or conditional shift.

Sample selection bias Covariate shift has some overlap with the sample selection bias problem (Heckman, 1979; Vella, 1998), which in statistics refers to the situation that the observed sample is not *i.i.d.* selected from the population. In fact, the sample selection bias problem reduces to covariate shift if we assume that the selection bias is independent of the labels of the data.

We can understand the sample selection bias as a special type of domain adaptation problems. Particularly, the plenty of labeled source data correspond to the observed and biased sample and the target domain corresponds to the population. A non-*i.i.d.* sampling process is assumed as follows using a selection variable $s \in \{0, 1\}$.

- An instance $(\boldsymbol{x}_i, y_i)$ is firstly drawn according to the target distribution $P_{\mathcal{T}}(\boldsymbol{x}, y)$.
- It is then put in the source domain with probability $P(s = 1|\boldsymbol{x}_i, y_i)$.

Assuming that the selection variable is independent of y, we have

$$P_{\mathcal{S}}(\boldsymbol{x}) \propto P_{\mathcal{T}}(\boldsymbol{x})P(s = 1|\boldsymbol{x}) \quad \text{and} \quad \omega(\boldsymbol{x}) = \frac{P_{\mathcal{T}}(\boldsymbol{x})}{P_{\mathcal{S}}(\boldsymbol{x})} \propto \frac{1}{P(s = 1|\boldsymbol{x})} \tag{2.11}$$

where the latter is the instance weight derived in eq. (2.4) under the ERM principle. Note that this implies that the conditional distribution $P(y|\boldsymbol{x})$ is shared by the source and the target domains, reducing to covariate shift. It is worth pointing out that, however, **Assumption 1** in Section 2.1.1 becomes more stringent since here the source and the target share the same support due to the sample selection process.

There has been a rich line of work on correcting the sample selection bias in statistics as early as 1970s (Heckman, 1979; Vella, 1998). In machine learning, perhaps Zadrozny's work (Zadrozny, 2004) was the first to examine how different classifiers could be affected by the sample bias. The analyses were then improved in (Fan et al., 2005).

To correct the sample bias, the above mentioned methods for covariate shift are in general applicable. Besides, the selection process actually hints a very simple method to estimate the instance weight through $\omega(\boldsymbol{x}) \propto 1/P(s = 1|\boldsymbol{x})$, where $P(s = 1|\boldsymbol{x})$ can be implemented as a discriminative classifier to classify the source sample $\{\boldsymbol{x}_m\}$ versus the target sample $\{\boldsymbol{x}_n\}$. This is utilized in (Bickel & Scheffer, 2007; Bickel et al., 2007). Ren *et al.* proposed to correct the bias locally, within the clusters of data (Ren et al., 2008). The idea is to cluster the data from both domains and use the target data to re-balance the biased source data in each cluster. Liao *et al.* introduced an auxiliary variable for each source data instance to offset the target classifier's decision value (Liao et al., 2005). These variables are learned together with the parameters of the classifiers. Beyond correcting the sample bias in learning classifiers, Dudík *et al.* studied it in maximum entropy density estimation (Dudík et al., 2005).

Target shift and conditional shift Zhang *et al.* examined some new assumptions in domain adaptation by drawing the motivation from causality (Zhang et al., 2013). One is called target shift. Namely, the marginal distributions of $P_{\mathcal{S}}(y)$ and $P_{\mathcal{T}}(y)$ are different, while the "causal" distributions $P(\boldsymbol{x}|y)$ are (almost) the same across the two domains. The other scenario is conditional shift. This refers to that the source and the target have different "causal" distributions $P_{\mathcal{S}}(\boldsymbol{x}|y)$ and $P_{\mathcal{T}}(\boldsymbol{x}|y)$, respectively, while $P(y)$ remains the same in both domains. The authors show that the target shift and conditional shift can be solved exactly for some families of distributions. The adaptation algorithm follows the instance weighting scheme we described in Section 2.1.1. It is worth noting that the target shift and conditional shift echo the causal learning in (Schölkopf et al., 2012). Besides, the target shift situation also occurs in learning from imbalanced data (Kubat & Matwin, 1997), where a few classes dominate the training set. Some

methods dealing with the imbalanced data could therefore be extended to solve the target shift problem, and vice versa.

2.1.4 Downside to instance weighting

We have seen that the covariate shift assumption leads to a natural solution, instance weighting, under the empirical risk minimization principle. Nonetheless, instance weighting does not always help domain adaptation.

If the true decision function $f : \mathcal{X} \mapsto \mathcal{Y}$ is realizable and is covered by the hypothesis set $\mathcal{H}$, there is no need of weighting the source instances. Instead, the optimal solution is to simply learn towards f solely on the source domain and then apply it to the target due to that the conditional distribution $P(y|\boldsymbol{x})$ is shared by the two domains. Some works try to formally formulate under which condition instance weighting is unnecessary (Shimodaira, 2000; Wen et al., 2014). The theory analyses (Cortes et al., 2008, 2010) have provided some generalization bounds and insights to this situation. However, it is still unclear how to detect such situation from real data.

The experimental results of instance weighting are mixed. As pointed out by Gretton *et al.* (Gretton et al., 2009), the MMD based instance weighting approach mainly improves the adaptation performance of simple models. It does not benefit high-capacity models too much. In other words, instance weighting is able to save the computation cost by letting one use simpler models (Storkey, 2009).

Instance weighting is not flexible in modeling the intrinsic structures in data. In some application domains, the data favor some special structures which could ease the adaptation of classifiers. For example, in NLP some predictive features are generic across different domains. In computer vision, the data often have low-rank or manifold properties. To exploit such structures, an alternative line of research works on learning feature representations for domain adaptation.

2.2 Feature learning approaches to domain adaptation

The high-level objective of learning features for domain adaptation is to map the input representations $\boldsymbol{x} \in \mathcal{X}$ to new feature representations $\boldsymbol{z} \in \mathcal{Z}$, such that 1) the source and the target domains are distributed about the same and 2) the discriminative information is preserved as much as possible — otherwise one may easily dump a new space $\mathcal{Z}$ satisfying 1) but being useless in terms of the classification tasks. Due to the scarce labels from the target domain, the mapping rarely relies on the labeling information of the target. In other words, the feature learning approach often implicitly involves the covariate shift assumption in the learned feature space; the learner tries to match $P_{\mathcal{S}}(\boldsymbol{z})$ with $P_{\mathcal{T}}(\boldsymbol{z})$ as much as possible while assuming that $P(y|\boldsymbol{z})$ is shared by the source and the target domains.

We next review some representative feature learning approaches. They exploit various *structures or properties* in the data of different domains to facilitate the adaptation.

2.2.1 Domain-generalizable versus domain-specific structures

In NLP tasks, for instance the part of speech tagging where we want to identify words as nouns, verbs, determiners, *etc.*, the word-based features often have the following property. Some features are strongly predictive (*e.g.*, "the" is a determiner) no matter in the source domain or in the target

domain, while others are more discriminative in one domain than in the other (*e.g.*, "monitor" likely implies a noun in IT product reviews and could be a verb in business magazines). This kind of structure also occurs in other application domains. In visual object recognition, the data often lies on a low-dimensional manifold or subspace. There are often overlaps between the underlying manifolds or subspaces of the source and target domains.

Therefore, a potential solution to domain adaptation is to identify those domain-generalizable feature representations and use them to build classifiers. However, this necessarily loses some discriminative power in the data. As shown in the empirical study (Blitzer et al., 2011) as well as our own experiments (*cf.* Table 9.1), the corresponding performance is about the same or only slightly better than no adaptation if we limit the classifiers to the scope of the domain-generalizable features only. A better strategy, which has been verified in many empirical studies, is to find the correspondence between the domain-specific and domain-generalizable structures, so that the correspondence information is used together with the generalizable features for training domain-invariant models.

Some of the existing works explicitly separate the domain-specific information from the generalizable structures before learning any correspondence between them. Blitzer *et al.* proposed to couple different views (subsets) of the features using canonical correlation analysis (CCA) (Blitzer et al., 2011). The source or target specific features are approximately estimated by their projections to the subspace which is orthogonal to the one shared by the two domains. Their approach is able to handle target features that have no support in the source domain at all. Li *et al.* used a generative model to generate documents using either shared or specific topics. The correlation matrix between shared and specific topics are therefore computable using the learned probabilistic distributions (Li et al., 2012). They map the domain-specific topics towards the shared ones using the correlation matrix. Pan *et al.* separated the shared features of two domains from the data by their appearing frequencies in the domains, their mutual information with the domain labels, and other heuristics (Pan et al., 2010). A bipartite graph is then constructed with edges connecting the shared features and the domain-specific ones. The correspondence is encoded as the Laplacian spectrum of the graph. A similar idea is explored in (Dai et al., 2009) by constructing graphs that encode more involved interplays among features; the graphs connect three kinds of nodes, data instances, features, and labels.

Alternatively, we can also implicitly model the correspondence between domain-generalizable and domain-specific features without actually separating them. In a surprisingly simple method, Daumé III (Daumé III, 2007) augmented the features of the source domain by $\boldsymbol{x}_m \mapsto [\boldsymbol{x}_m; \boldsymbol{x}_m; \mathbf{0}]$ and the features of the target domain by $\boldsymbol{x}_n \mapsto [\boldsymbol{x}_n; \mathbf{0}; \boldsymbol{x}_n]$. Consider a linear classifier. The second block of the augmented features encourages the classifier to use source-only information and the third block corresponds to the target-only information. Another popular form of mapping function is $\boldsymbol{x} \mapsto [\boldsymbol{x}, \Theta\boldsymbol{x}]$, where Θ takes the responsibility of correlating target-specific structure with the other features which are equally predictive to the class labels across domains.

Different methods have been proposed to learn Θ. One of the most popular methods is structural correspondence learning (SCL) (Blitzer et al., 2006). It follows the argument of (Ando & Zhang, 2005) and extends the method there to domain adaptation. The basic claim is that if some predictive structures (features) perform well in auxiliary tasks that are related with the main task, then those structures will very likely give rise to good performance on the main task as well. SCL uses "pivot" features to create several binary classification tasks. The mapping Θ is decomposed using SVD from the weights of those binary classifiers. Blitzer *et al.* improved SCL in (Blitzer

et al., 2007) by filtering out good pivot features using their mutual information with the class labels. Tan and Cheng further improved it by weighting the features and instances (Tan & Cheng, 2009). SCL also finds its applications in cross-language text classification (Prettenhofer & Stein, 2010; Margolis et al., 2010). In (Chen et al., 2009), Chen *et al.* learned Θ by minimizing the MMD (*cf.* Section 2.1.2) between two domains (However, they used a linear kernel, which is not characteristic, in MMD).

Discussion In general, the feature learning methods reviewed above outperform the instance weighting approaches according to the reported experimental results. The domain-generalizable and domain-specific features are easy to identify in some applications like NLP tasks. However, it becomes difficult to do so in other application domains like speech and computer vision; instead, implicitly modeling the correspondence by $\boldsymbol{x} \mapsto [\boldsymbol{x}; \Theta\boldsymbol{x}]$ is more appealing.

Note that both explicitly and implicitly modeling the correspondence result in linear transformations of the input space. Though one may wonder the domain discrepancy could be far more complicated than any linear transformations, the above reviewed methods are actually effective on real datasets. Besides, it is often straightforward to use kernel methods to introduce nonlinearity to the linear algorithms. The readers are referred to Section 3.1.3 for an instance.

Beyond the notions of domain-generalizable and domain-specific features, one can also directly learn a (linear) transformation to map the input $\mathcal{X}$ to a new feature space $\mathcal{Z}$, under the principle that in the new feature space the two domains should be statistically close and the discriminative capability of the data is preserved. This can be achieved by exploring other types of structures (low rank, cluster, sparsity, *etc.*) in the data.

2.2.2 Low-rank structure

The low-rank structure prevails in many applications. Most perceptual data (*e.g.*, images, speech signals, *etc.*) actually lie on low-dimensional manifolds or subspaces. In domain adaptation, it becomes effective to map the data to some low-dimensional spaces which minimize the statistical difference between two domains. In (Baktashmotlagh et al., 2013), Baktashmotlagh *et al.* learned a subspace projection (an orthonormal matrix) W such that the projected data $\boldsymbol{x} \mapsto W\boldsymbol{x}$ are similarly distributed across the domains. They minimize the MMD between the two domains using Gaussian RBF and polynomial kernels. An intra-class scatter term is included to take care of the discriminative information. Shu *et al.* exploited a locality-preserving embedding which is shared by the data from both domains (Shu et al., 2014). Guo and Xiao learned different subspaces for the two domains (Guo & Xiao, 2012). However, their method only considers the case that the source and target data are paired up (*e.g.*, a document and its translated version in another language). The low-rank structure could also be as simple as a subset of the features, as studied by Satpal and Sarawagi in the conditional random field (Satpal & Sarawagi, 2007). In a similar spirit, a dimension reduction algorithm is developed in (Wang et al., 2008).

The so called transfer component analysis (TCA) (Pan et al., 2009) learns a subspace of the empirical kernels, in contrast to the data in the original input space, by minimizing the domain discrepancy MMD and maximizing the data variance. Baktashmotlagh *et al.* proposed to measure two distributions on the statistical manifold (Baktashmotlagh et al., 2014). They showed better

experimental results than using the counterpart MMD. However, the resulting problem is non-convex and inefficient to solve for large-scale data. Fernando *et al.* found a mapping matrix between two domains by examining their respective subspaces (Fernando et al., 2013).

2.2.3 Sparsity structure

Sparse coding is a popular type of feature learning approaches (Aharon et al., 2006). It has been studied in and applied to various applications. The corresponding optimization algorithms are well-studied, making it a computationally appealing technique. Following the notations in (Aharon et al., 2006), let the example set be $\mathbf{Y} = [\mathbf{y}_1, \cdots, \mathbf{y}_N] \in \mathbb{R}^{n \times N}$. Sparse coding tries to find a dictionary $\mathbf{D} \in \mathbb{R}^{n \times K}$ and the sparse representations $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N] \in \mathbb{R}^{K \times N}$ of $\mathbf{Y}$, by solving the following problem,

$$\min_{\mathbf{D},\mathbf{X}} \{\|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2\} \quad \text{subject to} \quad \forall i, \|\mathbf{x}_i\|_0 \leq T_0, \tag{2.12}$$

where T_0 is the parameter controlling the sparsity of the new representations $\mathbf{X}$.

When it comes to domain adaptation, one would like to learn the representations of both domains such that the domain discrepancy is minimized under the new representation. The authors of (Long et al., 2013) introduced two regularizers upon the new representations to learn. One is a graph smooth term. The other is to match two domains by MMD. A third term, similar to the objective function of SVM, is introduced in (Shekhar et al., 2013). The overall formulation has to be solved alternatively. Zheng *et al.* tried to maximize the data variance and cross-domain correlation and meanwhile to learn the sparse coding (Zheng et al., 2013). Huang and Wang proposed a coupled dictionary learning approach for the domains which are paired up (Huang & Wang, 2013), the same setting as studied in (Guo & Xiao, 2012). Two coupled dictionaries are learned simultaneously and only the representations of the two domains are matched up as close as possible.

2.2.4 Averaging intermediate feature representations

Recently, a new line of approaches to domain adaptation achieves good performance on the object recognition tasks. They model the domain shift from the source domain to the target domain by some intermediate "phantom" domains. Say we represent by two subspaces the source and the target domains, respectively. One can then find the intermediate subspaces interpolating between those two. These interpolated subspaces can be regarded as the representations of the phantom domains. If we project a data instance $\boldsymbol{x}$ into all of these subspaces, the projected overall features are analogous to that of (Daumé III, 2007); the new feature representations contain information of the source, the target, and the intermediate domains as well. Therefore, they enforce the classifiers to take care of all such information, instead of biasing towards any particular domain. This subspace-based interpolation is employed in (Gopalan et al., 2011, 2014), and later applied to location classification from images (Gopalan, 2013). Since the new representations have a very high dimensionality, often a dimension reduction step is required before actually training the classifiers. In contrast, we develop a more efficient and effective kernel method to harvest an infinite number of intermediate phantom domains in Chapter 4. Zheng *et al.* also explored the kernel method (Zheng et al., 2012) but did not examine the discriminative subspaces (*e.g.*, learned

by partial least squares) as we do. An interesting extension was proposed by Shrivastava *et al.* in (Shrivastava et al., 2014). Considering that a single subspace might be inadequate to represent a whole domain, the authors model each class of the source domain by a subspace, which is then associated with a subspace from the target domain. The domain shift is then modeled by the parallel transportations from all the source subspaces to their corresponding target subspaces.

In addition to the subspace based interpolation, Ni *et al.* recursively minimized the reconstruction error and obtained a series of intermediate representations (Ni et al., 2013). Chopra *et al.* randomly selected different proportions of the data from the two domains to build intermediate domains (Chopra et al., 2013).

2.2.5 Other feature learning approaches to domain adaptation

A metric learning approach was modified from the information theoretic metric learning (Davis et al., 2007) to account for the mismatch between two domains (Saenko et al., 2010). It was later extended to a kernel version (Kulis et al., 2011). These approaches require a few labels from the target domain. We regard them as feature learning approaches since we can decompose the covariance matrix in the Mahalanobis distance and use it as a feature mapping function.

Most recently, deep learning has been applied in the domain adaptation problems and has achieved impressive performance (Glorot et al., 2011; Chopra et al., 2013; Donahue et al., 2014; Oquab et al., 2014). Chen *et al.* derived a closed-form method (Chen et al., 2012) from the deep auto-encoder. So far these approaches mainly rely on a sufficiently large and labeled dataset from the source domain. Indeed, in the ideal case if we had enough labeled data, there would be no domain adaptation issues at all since the labeled data could cover all the characteristics in the target domain. However, it remains questionable how the "deep" methods could be utilized when there are limited training data from the source domain.

In a broader setting, the methods aiming to learn robust classifiers can be readily applied to domain adaptation. There the main objective is to avoid overly training any particular features such that the classifier is robust to missing features or mismatched features in the test stage. Some popular techniques include re-weighting features (Kołcz & Teo, 2009; Dekel et al., 2010; Dredze et al., 2008) and injecting noises to the data (Globerson & Roweis, 2006; Teo et al., 2007).

2.3 Directly adapting models to the target domain

The objective of domain adaptation is to have a generative or discriminative model to perform well on the target domain. In the traditional supervised learning (*i.e.*, assuming the training and test data are *i.i.d.* drawn from the same underlying distribution), there are several well established and practically well performed models, such as SVMs, Boosting, hidden Markov models (HMMs), and so on. Depending on which models to use, one may derive different adaptation techniques to generalize these models to the target domain. Such approaches are often particularly tailored for the models (*e.g.*, SVMs) to be adapted. They are somehow "orthogonal" to the feature learning approaches, in the sense that one can either learn features before adapting the models or couple these two stages into a unified framework.

When dealing with probabilistic models, it is straightforward to take the model learned from the source domain as the prior for the target domain (Raina et al., 2006; Chelba & Acero, 2006; Dixit et al., 2011; Ciaramita & Chapelle, 2010; Reynolds et al., 2000). Sometimes the models

of the source domain are referred to as the "background" models. As for non-probabilistic approaches, often one has to fine-tailor the adaptation strategies according to the characteristics of the models for domain adaptation. Some representative methods include adapting SVMs (Duan et al., 2009a; Ma et al., 2013), Boosting (Dai et al., 2007b; Pardoe & Stone, 2010; Habrard et al., 2013), naive Bayes (Dai et al., 2007a; Tommasi & Caputo, 2013), and so on. Another popular idea is the so called "self-paced" learning. One can label the target data using the source classifiers. High-confident labels are then merged to the source domain to retrain the classifier. This process can be repeated many times until convergence (Chen et al., 2011b; Bruzzone & Marconcini, 2010; Tang et al., 2012; Bacchiani & Roark, 2003; Gaidon & Vig, 2014; Rosset et al., 2004). However, such methods have the risk of being contaminated by wrongly labeled data, which may be serious when the two domains differ to a large degree. One can also learn the models and feature representations simultaneously, such as the kernel-based SVM methods (Duan et al., 2012a, 2009a; Shi & Sha, 2012; Hoffman et al., 2013). Due to the coupled feature mapping parameters and the model parameters, the resultant formulations are in general non-convex. Alternative or EM-style optimization is thus required.

2.4 Some open questions and other domain adaptation scenarios

So far we have been focusing on the domain adaptation setting involving a single source domain and the target domain. Next, we discuss other related scenarios and some open questions in domain adaptation.

In real applications, we may have multiple labeled datasets as the potential source domains and some unlabeled data from the target domain. What is the best strategy to build a good classifier for the target domain in this case? One may go for the *multi-source adaptation* approaches (Mansour et al., 2009b; Duan et al., 2009b, 2012c). However, it is not necessarily the best strategy especially when there exists a source domain which is quite close to the target. Without running adaptation algorithms, however, can we infer which source domain is the best to be adapted to the target? To approach this problem, we provide a rank-of-domains (ROD) metric in Chapter 6. Yet another question is that, given many source domains but no data from any target domain, can we learn a robust classifier to generalize well to the future test data? This is termed as *domain generalization* and has been studied in (Huang & Yates, 2010; Muandet et al., 2013; Xu et al., 2014). It shares similar spirit as learning robust classifiers (Kołcz & Teo, 2009; Dekel et al., 2010; Dredze et al., 2008; Globerson & Roweis, 2006; Teo et al., 2007) except that we have the extra domain labels of the training data.

Domain knowlege or prior is often very useful in application-specific problems, such as face recognition and NLP. It greatly benefits the corresponding adaptation methods. We give several examples here: facial landmark localization (Smith & Zhang, 2014), human motion parsing from videos (Shen et al., 2014), speaker adaptation through HMMs (Leggetter & Woodland, 1995a), language model adaptation (Bacchiani & Roark, 2003), cross-language text classification via translation and adaptation (Shi et al., 2010), and so on.

When we talk about domain adaptation in this thsis, we assume that the domains share the same input and output spaces. In a broader setting, researchers have also examined the case that the domains have distinct input spaces, called heterogeneous domain adaptation (Argyriou et al., 2008; Duan et al., 2012b). How to systematically transfer knowledge from various sources to aid the learning of the target task is regarded as transfer learning. We refer the readers to the

survey paper (Pan & Yang, 2010) for a better understanding. Domain adaptation techniques can significantly benefit lifelong learning (Lazaric et al., 2013; Chen & Liu, 2014). They are also closely related with multi-task learning (Evgeniou & Pontil, 2007). Indeed, we can understand the semisupervised domain adaptation as a two-task learning problem. Domain adaptation is also aligned with active learning and few-shot learning in the sense that they all alleviate the need of many labeled data for the target task. In (Wang et al., 2014) and (Chattopadhyay et al., 2013) the authors combined the strengths of the adaptation and active learning methods, while one-shot learning can benefit from transfer learning shown in (Yu & Aloimonos, 2010; Socher et al., 2013).

2.5 Existing works closely related to this thesis

We discuss a few exiting methods that are closely related to ours.

Of particular relevance to our work is the idea of learning new feature representations that are domain-invariant, thus enabling almost effortlessly transferring classifiers from the source domain to the target domain (Ben-David et al., 2007; Blitzer et al., 2006, 2007; Daumé III, 2007; Pan et al., 2009). The feature representation can be derived using auxiliary tasks that predict "pivot features" shared across domains (Ando & Zhang, 2005; Blitzer et al., 2006), augmenting the feature space (Daumé III, 2007; Daumé III et al., 2010; Li & Zickler, 2012; Gopalan, 2013), co-training with multi-view representation (Chen et al., 2011a), or matching probabilistic distributions (Pan et al., 2009). Those approaches are especially appealing to unsupervised domain adaptation as they do not require labeled target data.

Gopalan *et al.*'s work (Gopalan et al., 2011) is the closest to our geodesic flow kernel (see Chapter 4). Both methods explore the idea of using geodesic flows to derive intermediate subspaces that interpolate between the source and target domains. However, a crucial difference between Gopalan *et al.*'s work and ours is the number of subspaces used to obtain the overall new feature representations. They sampled a *finite* number of subspaces and stacked these subspaces into a very high-dimensional projection matrix. As such, the dimension of their features needs to be reduced before one feeds them into a classifier. This extra step, unfortunately, might introduce modeling errors. It is not clear how to choose the sampling rate or the right dimension and whether the dimension reduction method helps or harms classification.

In stark contrast, our kernel method is both conceptually cleaner and computationally simpler, eliminating the need to tune many parameters. In particular, our kernel is in a closed form and computing it involves simple matrix algebra like singular value decomposition. Moreover, the closed-form nature of the kernel (*cf.* eq. (4.5)) indicates that the dimension of the new feature space is no more than the dimension of the original feature space—there is no dimensionality curse though we summing up infinitely many subspaces. We note that Zheng *et al.* proposed the same kernel, albeit independently and almost simultaneously (Zheng et al., 2012). Our approach, however, is able to automatically determine the dimensions of the subspaces and introduces discriminative subspaces to domain adaptation.

While learning domain-invariant feature representations has been extensively studied in the literature, identifying and using instances that are distributed similarly to the target to bridge the two domains, as in our landmark-based approach (see Chapter 5), has not been explored before.

The idea of auxiliary tasks was explored previously in (Blitzer et al., 2006) to identify invariant features, where the tasks were to predict "pivot features". The derived features are the PCA

directions of those predictive models' parameters. In our landmark-based approach, however, the auxiliary tasks are domain adaptation tasks and the invariant features are learned *discriminatively*.

The procedure of moving labeled data between the source and the target domains in our landmark-based approach shares some similarity to transductive domain adaptation methods (Bergamo & Torresani, 2010; Bruzzone & Marconcini, 2010; Chen et al., 2011a) where classifiers are iteratively retrained. There are several major properties setting our approach apart from them. We partition the source domain into two disjoint subsets only once for each auxiliary task. We do not iteratively re-partition the datasets, whereas in those methods the target and the source domains are merged iteratively. Besides, our primary goal is to learn useful features from the auxiliary tasks and we treat each auxiliary task as an unsupervised domain adaptation problem. In contrast, those methods use the original features and aim to directly adapt the classifiers.

MMD has previously been used to weight data instances from the source domain (Huang et al., 2006; Pan et al., 2009; Gretton et al., 2009) to correct the mismatch between the two domains, while we *select* instances as our landmarks using MMD. Additionally, the new class balancing constraint in our formulation of eq. (5.4) is crucial, as evidenced in our comparison to other methods in the experimental studies (*cf.* Table 9.3).

Chapter 3

Kernel Methods: A Gentle Tutorial

Surprisingly or not, many statistical machine learning algorithms depend on only inner products between data points. These include but are not limited to support vector machines (SVMs), Ridge regression, Fisher's linear discriminant analysis (Fisher's LDA), principal component analysis (PCA), K-means, and spectral clustering. Kernel methods are flexible techniques to expand the horizon of such methods from at least four perspectives.

- A kernel function implicitly induces a feature mapping, which may not be unique, from the original data space to a new space. Such mapping can be nonlinear and the dimensionality of the new space can be virtually infinite. As a result, kernels are often used to define *nonlinear* decision boundaries, correlations, cluster boundaries, and other properties of data in the algorithms of SVMs, LDA, PCA, and so on. Besides, it is often computationally much more efficient to play with kernels than to work with the induced feature mappings. We shall see this point more clearly in our GFK approach in Chapter 4.

- A kernel can be a composition of several other kernels. As a result, kernel methods, for instance multiple kernel learning, provides a clean and versatile framework for combining heterogeneous data sources, or different representations of the same dataset, to jointly make decisions or inferences. To this end, we can define one or more kernels from each type of the data sources and then specify or learn some mechanism to combine these kernels into a new kernel. One of such techniques is used in our landmark based method in Chapter 5.

- It is convenient to design kernels taking as input symbolic data (sequences, trees, graphs, *etc.*). One can thus readily extend the above mentioned machine learning algorithms from dealing with vectors to working with more other data formats.

- Kernels enable discriminative methods to take advantage of the complementary generative modeling approaches. Consider a supervised learning problem. Generative models specify a joint distribution $P(\boldsymbol{x}, y)$ over the observation variables $\boldsymbol{x}$ and target variable y, while discriminative methods only model the dependency of the target variable on the observation variables, or $P(y|\boldsymbol{x})$, under the probabilistic framework. Comparing to the discriminative methods, the former is able to handle missing data and to capture complex dependencies among all variables. These two types of approaches are mutually complementary and we can seamlessly connect them by using kernels.

Next, we provide a gentle tutorial to the kernel methods and focus on the well-known "kernel trick" (Section 3.1) and the kernel embedding of distributions (Section 3.2), both of which are

used in the following chapters. For a more comprehensive understanding of the kernels please be referred to the textbooks (Schölkopf & Smola, 2002; Shawe-Taylor & Cristianini, 2004; Herbrich, 2002; Kung, 2014).

3.1 Dual representation and the kernel trick

It is convenient to motivate kernel methods by showing that many machine learning algorithms depend on only inner products between data points. To this end, we introduce the notion of dual representation, followed by the statement of the kernel trick. Moreover, we give an illustrating example, kernel PCA, at the end of this section.

3.1.1 Dual representation

Consider a training set $S = \{(\boldsymbol{x}_m, y_m)\}_{m=1}^{\mathsf{M}}$ and a linear function

$$f(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b, \tag{3.1}$$

which shows up in SVM, Logistic regression, Fisher's LDA, PCA, *etc.* The model parameter $\boldsymbol{w}$ can be decomposed into two components,

$$\boldsymbol{w} = \sum_{m=1}^{\mathsf{M}} \alpha_m \boldsymbol{x}_m + \boldsymbol{w}^{\perp}, \quad \alpha_m \in \mathbb{R}, m = 1, 2, \cdots, \mathsf{M} \tag{3.2}$$

where the first lies in the span of the training set S and the latter is the part complementary to the span. When learning the parameter $\boldsymbol{w}$ using the training set, we usually optimize with respect to the function values of the observed data,

$$f(\boldsymbol{x}_j) = \sum_{m=1}^{\mathsf{M}} \alpha_m \langle \boldsymbol{x}_m, \boldsymbol{x}_j \rangle + \left\langle \boldsymbol{w}^{\perp}, \boldsymbol{x}_j \right\rangle + b = \sum_{m=1}^{\mathsf{M}} \alpha_m \langle \boldsymbol{x}_m, \boldsymbol{x}_j \rangle + b, \quad \forall \boldsymbol{x}_j \in S \tag{3.3}$$

where $\left\langle \boldsymbol{w}^{\perp}, \boldsymbol{x}_j \right\rangle = 0$ for all $\boldsymbol{x}_j \in S$. In other words, we can actually express the model parameter $\boldsymbol{w}$ by a linear combination of the observed data without changing its behavior on the sample S,

$$\boldsymbol{w} = \sum_{m=1}^{\mathsf{M}} \alpha_m \boldsymbol{x}_m, \quad \alpha_m \in \mathbb{R}, m = 1, 2, \cdots, \mathsf{M} \tag{3.4}$$

which we call the *dual representation* of $\boldsymbol{w}$ in the linear function $f(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b$.

Given an arbitrary input $\boldsymbol{x}$, we can see from eq. (3.3) that the linear function $f(\boldsymbol{x})$ depends on only the inner products between $\boldsymbol{x}$ and the training instances in S under the dual representation. It is not difficult to validate this for some statistical learning algorithms by the duality technique in the mathematical optimization theory. Indeed, one can easily find the corresponding derivations of SVMs and Ridge regression from some textbooks. Instead of ruminating them again, we will give another illustrating example in Section 3.1.3, kernel PCA (Schölkopf et al., 1997), in which we shall arrive at the dual representation through plain linear algebra.

For a more formal treatment of the dual representation described above please see the representer theorems (Kimeldorf & Wahba, 1970; Schölkopf et al., 2001; Schölkopf & Smola, 2002).

3.1.2 The kernel trick

Denote by $\mathcal{X}$ the input space of the data. If we apply some sort of feature transformation $\phi : \mathcal{X} \mapsto \mathcal{H}$ and work in the embedding space $\mathcal{H}$, we have

$$f(\phi(\boldsymbol{x})) = \sum_{m=1}^{\mathsf{M}} \alpha_m \langle \phi(\boldsymbol{x}_m), \phi(\boldsymbol{x}) \rangle + b, \quad \forall \boldsymbol{x} \in \mathcal{X} \tag{3.5}$$

i.e., the inner products $k(\boldsymbol{x}, \boldsymbol{z}) \triangleq \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{z}) \rangle$ in the embedding space $\mathcal{H}$ are sufficient for computing $f(\phi(\boldsymbol{x}))$. There is virtually no need of the explict mapping function $\phi(\cdot)$ under the dual representation as long as we can calculate the inner product $k(\boldsymbol{x}, \boldsymbol{z})$, which is actually a kernel function over the original feature space $\mathcal{X}$.

Definition 1. *A function $k : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ is called a* kernel *over $\mathcal{X}$.*

It is often computationally cheaper and more flexible to play with the kernel function than to work with the feature mapping $\phi(\cdot)$. This approach is called the "kernel trick", which we leverage to derive our geodesic flow kernel in Chapter 4. The kernel trick is feasible thanks to Mercer's theorem, which guarantees the existence of $\phi(\cdot)$ for a rich set of kernels.

Theorem 1. *(Mercer's) A symmetric kernel $k(\boldsymbol{x}, \boldsymbol{z})$ can be expressed as an inner product*

$$k(\boldsymbol{x}, \boldsymbol{z}) = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{z}) \rangle \tag{3.6}$$

for some $\phi(\cdot)$ if and only if $k(\boldsymbol{x}, \boldsymbol{z})$ is symmetric positive definite (SPD)*, i.e.,*

$$\int c(\boldsymbol{x}) k(\boldsymbol{x}, \boldsymbol{z}) c(\boldsymbol{z}) \mathrm{d}\boldsymbol{x} \mathrm{d}\boldsymbol{z} \geq 0, \tag{3.7}$$

for any square integrable function $c(\cdot) \in L_2(\mathcal{X})$, or equivalently, the Gram matrix

$$K = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & k(\boldsymbol{x}_1, \boldsymbol{x}_2) & \cdots \\ k(\boldsymbol{x}_2, \boldsymbol{x}_1) & k(\boldsymbol{x}_2, \boldsymbol{x}_2) & \cdots \\ \vdots & \ddots & \end{bmatrix} \succeq 0 \tag{3.8}$$

is positive semidefinite for any collection $\{\boldsymbol{x}_m \in \mathcal{X}, m = 1, 2, \cdots\}$.

In the remaining of this thesis, we always assume that the Mercer's condition holds when we talk about kernels. We conclude this subsection by introducing the concept of reproducing kernel Hilbert space (RKHS).

Theorem 2. *(Moore-Aronszajn) For an SPD kernel $k(\cdot, \cdot)$ over $\mathcal{X}$, there exists a unique Hilbert space $\mathcal{H}$ of functions on $\mathcal{X}$ satisfying the following* reproducing property*,*

$$\forall h \in \mathcal{H}, \quad \forall \boldsymbol{x} \in \mathcal{X}, \qquad h(\boldsymbol{x}) = \langle h, k(\boldsymbol{x}, \cdot) \rangle . \tag{3.9}$$

Further, $\mathcal{H}$ is called the reproducing kernel Hilbert space (RKHS) associated to $k(\cdot, \cdot)$.

3.1.3 Kernel PCA: an illustrating example

The kernel trick enables us to perform efficient PCA in the embedding space $\mathcal{H}$, which could be high-dimensional or even have an infinite number of dimensions depending on the (nonlinear) feature mapping $\phi(\cdot)$. We demonstrate the power of the "trick" using kernel PCA. To this end, we begin working with $\phi(\cdot)$, show that PCA only depends on inner products between sample points, and finally replace the inner products by the kernels. One can use the same procedure to "kernelize" SVMs, Ridge regression, Fisher's LDA, canonical correlation analysis, *etc.* and your own learning algorithms.

Without loss of generality, assume that our observed data $S = \{(\boldsymbol{x}_m, y_m)\}_{m=1}^{\mathsf{M}}$ in the embedding space have been zero-centered, *i.e.*, $\sum_{m=1}^{\mathsf{M}} \phi(\boldsymbol{x}_m) = \mathbf{0}$. The covariance matrix is then computed by

$$C = \frac{1}{\mathsf{M}} \sum_{m=1}^{\mathsf{M}} \phi(\boldsymbol{x}_m)\phi(\boldsymbol{x}_m)^T. \tag{3.10}$$

The principal components correspond to the eigenvectors of the covariance matrix

$$\lambda \boldsymbol{v} = C\boldsymbol{v} = \frac{1}{\mathsf{M}} \sum_{m=1}^{\mathsf{M}} \phi(\boldsymbol{x}_m)\phi(\boldsymbol{x}_m)^T \boldsymbol{v} \tag{3.11}$$

with the constraint $\langle \boldsymbol{v}, \boldsymbol{v} \rangle = 1$. Of the most interest among all the eigenvectors are those whose corresponding eigenvalues are non-zero, or more precisely those with $\lambda > 0$. We therefore claim that the dual representations of such eigenvectors hold, *i.e.*,

$$\boldsymbol{v} = \sum_{m=1}^{\mathsf{M}} \alpha_m \phi(\boldsymbol{x}_m), \quad \alpha_m \in \mathbb{R}, m = 1, \cdots, \mathsf{M}, \tag{3.12}$$

where the coefficients $\alpha_m = \frac{1}{\lambda \mathsf{M}} \langle \phi(\boldsymbol{x}_m), \boldsymbol{v} \rangle$.

The dual representation of $\boldsymbol{v}$ tells that it lies in the span of the observed data (after the mapping $\phi(\cdot)$). The eigenvalue equation $\lambda \boldsymbol{v} = C\boldsymbol{v}$ is thus equivalent to the following,

$$\langle \phi(\boldsymbol{x}_j), \lambda \boldsymbol{v} \rangle = \langle \phi(\boldsymbol{x}_j), C\boldsymbol{v} \rangle = \left\langle \phi(\boldsymbol{x}_j), \frac{1}{\mathsf{M}} \sum_{m=1}^{\mathsf{M}} \phi(\boldsymbol{x}_m)\phi(\boldsymbol{x}_m)^T \boldsymbol{v} \right\rangle, \quad \forall \boldsymbol{x}_j \in S. \tag{3.13}$$

Substituting eq. (3.12) to (3.13), we have

$$\left\langle \phi(\boldsymbol{x}_j), \lambda \sum_{m=1}^{\mathsf{M}} \alpha_m \phi(\boldsymbol{x}_m) \right\rangle = \left\langle \phi(\boldsymbol{x}_j), \frac{1}{\mathsf{M}} \sum_{n=1}^{\mathsf{M}} \phi(\boldsymbol{x}_n)\phi(\boldsymbol{x}_n)^T \sum_{m=1}^{\mathsf{M}} \alpha_m \phi(\boldsymbol{x}_m) \right\rangle, \quad \forall \boldsymbol{x}_j \in S.$$

Denote by $K_{jm} \triangleq k(\boldsymbol{x}_j, \boldsymbol{x}_m) = \langle \phi(\boldsymbol{x}_j), \phi(\boldsymbol{x}_m) \rangle$, the above equation reads

$$\lambda \sum_{m=1}^{\mathsf{M}} \alpha_m K_{jm} = \frac{1}{\mathsf{M}} \sum_{n=1}^{\mathsf{M}} \sum_{m=1}^{\mathsf{M}} \alpha_m K_{jn} K_{nm}, \quad \forall j \in \{1, 2, \cdots, \mathsf{M}\} \tag{3.14}$$

or, in the matrix form, $\lambda \mathsf{M} K\boldsymbol{\alpha} = K^2\boldsymbol{\alpha}$, where $\boldsymbol{\alpha} = [\alpha_1, \cdots, \alpha_{\mathsf{M}}]^T$.

Similarly, we have the following.

- The constraint $\langle \boldsymbol{v}, \boldsymbol{v} \rangle = 1$ translates to $\boldsymbol{\alpha}^T K\boldsymbol{\alpha} = 1$.
- The encoding operation $\langle \boldsymbol{v}, \phi(\boldsymbol{x}) \rangle$ becomes $K(\boldsymbol{x}, \cdot)\boldsymbol{\alpha}$, where $K(\boldsymbol{x}, \cdot) = [k(\boldsymbol{x}, \boldsymbol{x}_1), \cdots, k(\boldsymbol{x}, \boldsymbol{x}_{\mathsf{M}})]$.

Unfortunately, the decoding/reconstruction operation of PCA in the embedding space cannot be done through kernels, because the kernels essentially hide the "real" encoded images by avoiding the explicit mapping $\phi(\cdot)$.

We thus have transformed the PCA problem in the embedding space, related to the original space by the (nonlinear) feature mapping $\phi(\cdot)$, to a new eigenvalue problem defined through the kernel $k(\cdot, \cdot)$ over the original feature space,

$$\lambda \mathsf{M} K\boldsymbol{\alpha} = K^2\boldsymbol{\alpha} \quad \text{s.t.} \quad \boldsymbol{\alpha} K\boldsymbol{\alpha} = 1 \tag{3.15}$$

$$\Leftrightarrow \quad \lambda \mathsf{M}\boldsymbol{\alpha} = K\boldsymbol{\alpha} \quad \text{s.t.} \quad \boldsymbol{\alpha} K\boldsymbol{\alpha} = 1. \tag{3.16}$$

One can specify any kernel $k(\cdot, \cdot)$ (which must satisfy Mercer's condition) and then solve the above eigenvalue problem for the coefficients $\boldsymbol{\alpha}$. This set of techniques is named kernel PCA (Schölkopf et al., 1997). Note that we arrive at kernel PCA by starting from the standard PCA in the embedding space and then replace all inner products by the kernel $k(\cdot, \cdot)$. This procedure is generic, being applicable to many algorithms. The kernel trick is thus also called kernel substitution under some contexts.

3.1.4 Popular kernels

We review several popular kernels in this section.

- Linear kernel: $k(\boldsymbol{x}, \boldsymbol{z}) = \langle \boldsymbol{x}, \boldsymbol{z} \rangle$
- Polynomial kernel: $k(\boldsymbol{x}, \boldsymbol{z}) = (\langle \boldsymbol{x}, \boldsymbol{z} \rangle + c)^n, c \geq 0, n \in \mathbb{N}$
- Exponential kernel: $k(\boldsymbol{x}, \boldsymbol{z}) = \exp(\gamma \langle \boldsymbol{x}, \boldsymbol{z} \rangle), \gamma \geq 0$
- Gaussian kernel: $k(\boldsymbol{x}, \boldsymbol{z}) = \exp(-\gamma \|\boldsymbol{x} - \boldsymbol{z}\|^2), \gamma \geq 0$
- Laplacian kernel: $k(\boldsymbol{x}, \boldsymbol{z}) = \exp(-\gamma \|\boldsymbol{x} - \boldsymbol{z}\|), \gamma \geq 0$
- Inverse distance kernel: $k(\boldsymbol{x}, \boldsymbol{z}) = \frac{1}{\|\boldsymbol{x} - \boldsymbol{z}\| + \gamma}, \gamma \geq 0$
- Histogram intersection kernel:

$$k(\boldsymbol{x}, \boldsymbol{z}) = \sum_{d=1}^{\mathsf{D}} \min(x_d, z_d),$$

where $\boldsymbol{x}$ and $\boldsymbol{z}$ are D-dimensional histograms $\{\boldsymbol{x} \in \mathbb{R}^{\mathsf{D}} \mid x_d \geq 0, \sum_{d=1}^{\mathsf{D}} x_d = 1\}$.

- χ^2 kernel:

$$k(\boldsymbol{x}, \boldsymbol{z}) = 1 - \sum_{d=1}^{\mathsf{D}} \frac{(x_d - z_d)^2}{2(x_d + z_d)} = \sum_{d=1}^{\mathsf{D}} \frac{2x_d z_d}{(x_d + z_d)},$$

  where $\boldsymbol{x}$ and $\boldsymbol{z}$ are D-dimensional histograms $\{\boldsymbol{x} \in \mathbb{R}^{\mathsf{D}} \mid x_d \geq 0, \sum_{d=1}^{\mathsf{D}} x_d = 1\}$.

- Fisher kernel (Jaakkola et al., 1999): The Fisher kernel is popular for exploiting generative models in discriminative (kernelizable) methods. It measures the pairwise similaries between data points induced by a generative model $P(\boldsymbol{x}; \boldsymbol{\theta})$. Particularly, the kernel is defined over the Fisher score vector $g_{\boldsymbol{\theta}}(\boldsymbol{z}) = \nabla \ln P(\boldsymbol{z}; \boldsymbol{\theta})$,

$$k(\boldsymbol{x}, \boldsymbol{z}) = g_{\boldsymbol{\theta}}(\boldsymbol{x})^T F^{-1} g_{\boldsymbol{\theta}}(\boldsymbol{z})$$

  where $F = \mathbb{E}_{\boldsymbol{x}} \left[g_{\boldsymbol{\theta}}(\boldsymbol{x}) g_{\boldsymbol{\theta}}(\boldsymbol{x})^T \right]$ is the Fisher information matrix. In practice the Fisher information matrix is usually computed empirically or simply set as the identity matrix.

- Kernels for matching sets of features: The so called data point $\boldsymbol{x}$ could be a set of features. There have been various kernels proposed to match the sets. The pyramid match kernel (Grauman & Darrell, 2005) and its extension, spatial pyramid matching (Lazebnik et al., 2006), have been extensively used in image classification. The Bhattacharyya affinity between two Gaussian distributions can be calculated in a closed form when one uses MLE to estimate the parameters of the Gaussian distributions. The affinity is regarded as a kernel (Kondor & Jebara, 2003). A more generic framework of explointing the sets of features via generative models is to define the kernel as

$$k(\boldsymbol{x}, \boldsymbol{z}) = \int P(\boldsymbol{x}|\boldsymbol{\pi}) P(\boldsymbol{z}|\boldsymbol{\pi}) P(\boldsymbol{\pi}) \mathrm{d}\boldsymbol{\pi}$$

  where $\boldsymbol{\pi}$ are the hidden states in for instance Hidden Markov models.

- Other kernels: we refer the readers to Part III of the book (Shawe-Taylor & Cristianini, 2004) for some representative kernels for texts, sequences, trees, and other types of structured data.

It often requires domain knowledge and experimental studies to choose the right kernels for a specific application. We can also composite new kernels from simple ones. Given valid kernels $k_1(\cdot, \cdot)$ and $k_2(\cdot, \cdot)$, the following new kernels are still valid, meaning that they satisfy Mercer's condition.

1. $k(\boldsymbol{x}, \boldsymbol{z}) = c k_1(\boldsymbol{x}, \boldsymbol{z}), \quad c \geq 0$
2. $k(\boldsymbol{x}, \boldsymbol{z}) = c(\boldsymbol{x}) k_1(\boldsymbol{x}, \boldsymbol{z}) c(\boldsymbol{z}), \quad c : \mathcal{X} \mapsto \mathbb{R}$
3. $k(\boldsymbol{x}, \boldsymbol{z}) = q(k_1(\boldsymbol{x}, \boldsymbol{z})), \quad q(\cdot)$: a polynomial with non-negative coefficients
4. $k(\boldsymbol{x}, \boldsymbol{z}) = \exp(k_1(\boldsymbol{x}, \boldsymbol{z}))$
5. $k(\boldsymbol{x}, \boldsymbol{z}) = k_1(\boldsymbol{x}, \boldsymbol{z}) + k_2(\boldsymbol{x}, \boldsymbol{z})$
6. $k(\boldsymbol{x}, \boldsymbol{z}) = k_1(\boldsymbol{x}, \boldsymbol{z}) k_2(\boldsymbol{x}, \boldsymbol{z})$

7. $k(\boldsymbol{x}, \boldsymbol{z}) = k_3(\phi(\boldsymbol{x}), \phi(\boldsymbol{z})), \quad \phi : \mathcal{X} \mapsto \mathbb{R}^{\mathsf{L}}, k_3(\cdot, \cdot)$: a valid kernel over $\mathbb{R}^{\mathsf{L}}$

8. $k(\boldsymbol{x}, \boldsymbol{z}) = \boldsymbol{x}^T \boldsymbol{A} \boldsymbol{z}, \quad \boldsymbol{A} \in \mathbb{S}_+$ is symmetric positive semidefinite

9. $k(\boldsymbol{x}, \boldsymbol{z}) = k_a(\boldsymbol{x}_a, \boldsymbol{z}_a), \quad \boldsymbol{x}_a$: a subset of the variables of $\boldsymbol{x}$, $k_a(\cdot, \cdot)$: a valid kernel over the space of $\boldsymbol{x}_a$

3.2 Kernel embedding of distributions

We present some major results of the kernel embedding of distributions summarized from (Smola et al., 2007; Gretton et al., 2006; Sriperumbudur, 2010). The embedding will appear in our approaches to domain adaptation in Chapter 5 and Chapter 7.

Recall some notations firstly. Denote by $\mathcal{X}$ the input space from which we have a sample $X = \{\boldsymbol{x}_m, m = 1, \cdots, \mathsf{M}\}$ drawn *i.i.d.* according to the distribution $P(\cdot)$ over $\mathcal{X}$, and by $\mathcal{H}$ the RKHS of a kernel $k : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$. The embedding of $P(\cdot)$ is then defined as

$$\mu[P] \triangleq \mathbb{E}_{\boldsymbol{x} \sim P}[k(\boldsymbol{x}, \cdot)] \tag{3.17}$$

$$\mu[X] \triangleq \frac{1}{\mathsf{M}} \sum_{m=1}^{\mathsf{M}} [k(\boldsymbol{x}_m, \cdot)] \tag{3.18}$$

where the latter is the empirical estimator of the former. Note that both $\mu[P]$ and $\mu[X]$ are respectively elements of the RKHS $\mathcal{H}$.

We can see that the embedding of the distribution P is nothing but the kernel mean calculated under P. Its significance is summarized by the theorem stated in (Fukumizu et al., 2004; Gretton et al., 2006).

Theorem 3. *The kernel mean map $\mu : \mathcal{P}_{\mathcal{X}} \mapsto \mathcal{H}$ of eq. (3.17) is injective if the kernel $k(\cdot, \cdot)$ is universal, where $\mathcal{P}_{\mathcal{X}}$ denotes the collection of probability measures on $\mathcal{X}$.*

We can therefore define a metric between two distributions by their respective kernel means in $\mathcal{H}$, $\mathrm{dist}(P_1, P_2) \triangleq \|\mu[P_1] - \mu[P_2]\|$, which is exactly the maximum mean discrepancy (MMD) we described in eq. (2.8). Moreover, it is straightforward to approximate this metric by the observations X_1 and X_2 respectively of the two distributions, $\widehat{\mathrm{dist}(P_1, P_2)} \triangleq \|\mu[X_1] - \mu[X_2]\|$. One thus does not need to know or to estimate the distributions in order to approximate the MMD distance between them. This is a significant result as it is often the case that we do not have the access to the underlying distributions of the data. However, one may wonder how good the approximation is by using the empirical kernel mean (eq. (3.18)). A bound between MMD and its empirical estimation is given in (Gretton et al., 2006) depending on the properties of the kernel and the sample size.

The choice of the kernel $k(\cdot, \cdot)$ is essential. As stated in Theorem 3, one has to use universal kernels to have a valid metric between distributions.

Definition 2. *(Universal kernel (Steinwart, 2002a)) A continuous kernel $k(\cdot, \cdot)$ on a compact metric space $(\mathcal{X}; d)$, where d stands for a metric on the compact input space $\mathcal{X}$, is called* universal *if the space of all functions induced by $k(\cdot, \cdot)$ is dense in $C(\mathcal{X})$, i.e., for every function $f \in C(\mathcal{X})$ and every $\epsilon > 0$ there exists a function g induced by k such that $\|f - g\|_\infty \leq \epsilon$.*

The Gaussian RBF kernel, Laplacian kernel, and the exponential kernel are universal. Steinwart provides several conditions to check whether or not a kernel is universal (Steinwart, 2002a). The notion of being universal is further extended to being characteristic in (Fukumizu et al., 2007) in the context of distribution embedding. Please be referred to (Sriperumbudur et al., 2010) for a great comparison between universal kernels and characteristic kernels.

Part II

Unsupervised Domain Adaptation with Kernel Methods

Chapter 4

Geodesic Flow Kernel

In this chapter, we introduce the geodesic flow kernel (GFK) for unsupervised domain adaptation. The kernel function measures similarity between data points in a domain-invariant feature space, thus enabling the adaptation of classifiers trained on the source domain readily to the target domain. We describe the main idea first, followed by the details on deriving the kernel. Our approach is particularly appealing as it is free of (hyper)parameter-tuning which often requires computationally intensive cross-validation; we describe how to choose automatically the only hyper-parameter in section 4.4.

4.1 Main idea

Our approach follows broadly the theme of identifying the *shared representation* between different domains (Ben-David et al., 2007). Intuitively, we seek a feature space such that when data points are projected into this space, the source domain is similar to the target domain.

How to define and quantify shared characteristics entails careful examination of our intuition on what type of representation facilitates adaptation. For example, in the part-of-speech (POS) task of tagging words into different syntactic categories (Blitzer et al., 2006), the idea is to extract shared patterns from auxiliary classification tasks that predict "pivot features", frequent words which are themselves discriminative in both domains. While sensible for language processing tasks, typical histogram based features of low-level visual or speech descriptors do not have the benefits of pivot " words" — in general, no single feature dimension from a particular histogram bin is discriminative enough to differentiate categories.

On the other hand, many perceptual data (*e.g.*, visual and speech data) are assumed to lie in low-dimensional subspaces. Given data from two domains, *how can we exploit the subspaces in these datasets, which can be telltale cues in revealing the underlying difference and commonness between the domains?*

The main idea behind our approach is to implicitly construct an infinite-dimensional feature space $\mathcal{H}^\infty$. This feature space assembles and aggregates information on the source domain $\mathcal{S}$, on the target domain $\mathcal{T}$, and on "phantom" domains interpolating between those two. In particular, the phantom domains represent incremental changes in the geometric and statistical properties between the two domains. While each of the domains is represented with a subspace, the inner products in $\mathcal{H}^\infty$ are defined to integrate over an infinite number of such subspaces. Intuitively, this integration averages out domain-specific idiosyncrasies and computes similarity measures that are insensitive to domain mismatch. Equivalently, the inner products give rise to a kernel

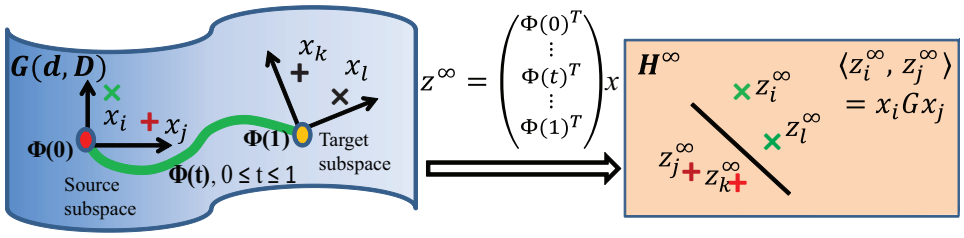

Figure 4.1: Main idea of our geodesic flow kernel-based approach for domain adaptation (best viewed in color). We embed source and target datasets in a Grassmann manifold. We then construct a geodesic flow (path) between the two points and integrate an infinite number of subspaces along the flow $\mathbf{\Phi}(t)$. Concretely, raw features are projected into these subspaces to form an infinite-dimensional feature vector $\boldsymbol{z}^\infty \in \mathcal{H}^\infty$. Inner products between these feature vectors define a kernel function that can be computed over the original feature space in closed-form. The kernel encapsulates incremental changes between subspaces that underly the difference and commonness between the two domains. The learning algorithms thus use this kernel to derive low-dimensional representations that are invariant to the domains.

function that defines the kernel mapping from the original feature space to a domain-invariant feature space.

In the following, we start by reviewing some basic notions of Grassmann manifolds; the subspaces of the data from the source and target domains are represented as two points on one such manifold. Furthermore, the phantom domains correspond to the points on the geodesic path connecting those two points. Fig. 4.1 sketches the main idea.

4.2 Modeling domains on Grassmann manifold

In statistical modeling, we often assume data can be embedded in a low-dimensional linear subspace. For example, principal component analysis (PCA) identifies the subspace where the variances of the embedded data are maximized. Most of the time, it is both sufficient and convenient to refer to a subspace with its basis $\boldsymbol{P} \in \mathbb{R}^{\mathsf{D}\times\mathsf{d}}$, where D is the dimensionality of the data and d is the dimensionality of the subspace. For PCA, the basis is then the top d eigenvectors of the data's covariance matrix. The collection of all d-dimensional subspaces form the Grassmannian $\mathbb{G}(\mathsf{d}, \mathsf{D})$, a smooth Riemannian manifold on which we can define geometric, differential, and probabilistic structures.

As an intuitive example of how manifolds can help us to attack the problem of domain adaptation, imagine that we compute the subspaces of the datasets for the $\mathcal{S}$ and $\mathcal{T}$ domains and map them to two points on a Grassmannian. Intuitively, if these two points are close by, then the two domains could be similar to each other; for example, their features may be similarly distributed. Thus, a $\mathcal{S}$-trained classifier is likely to work well on $\mathcal{T}$.

However, *what if these two domains are far apart on the manifold?* For example, suppose two datasets of car images with large differences in poses are placed far apart on the manifold.

We aim to use intermediate subspaces to learn domain-invariant features for adaptation[1]. Specifically, the intermediate subspaces would capture statistics of car images under poses interpolated between the source and the target domain. Being informed of all these different subspaces from the same category, the learning algorithms might be able to extract features that are less sensitive to variations in pose. To this end, we will use the geodesic flow path to connect the two domains, where every point on this flow path is an intermediate subspace.

4.3 Defining the geodesic flow kernel (GFK)

Our approach consists of the following steps: i) determine the optimal dimensionality of the subspaces to embed domains; ii) construct the geodesic flow path; iii) compute the geodesic flow kernel; iv) use the kernel to construct a classifier with labeled data. We defer describing step i) to the next section and focus on steps ii) and iii).

For step ii), we state only the main computational steps. The detailed derivation can be found in (Gopalan et al., 2011) and references therein. We also omit step iv) for brevity, as it is the same as constructing any other kernel-based classifier.

4.3.1 Construct geodesic flow

Let $\boldsymbol{P}_{\mathcal{S}}, \boldsymbol{P}_{\mathcal{T}} \in \mathbb{R}^{\mathsf{D}\times\mathsf{d}}$ denote the two sets of basis of the subspaces for the source and target domains. Let $\boldsymbol{R}_{\mathcal{S}} \in \mathbb{R}^{\mathsf{D}\times(\mathsf{D}-\mathsf{d})}$ denote the orthogonal complement to $\boldsymbol{P}_{\mathcal{S}}$, namely $\boldsymbol{R}_{\mathcal{S}}^{\mathrm{T}}\boldsymbol{P}_{\mathcal{S}} = \boldsymbol{0}$. Using the canonical Euclidean metric for the Riemannian manifold, the geodesic flow is parameterized as $\boldsymbol{\Phi} : t \in [0,1] \rightarrow \boldsymbol{\Phi}(t) \in G(\mathsf{d},\mathsf{D})$ under the constraints that $\boldsymbol{\Phi}(0)$ be the subspace of the source domain and $\boldsymbol{\Phi}(1)$ be the subspace of the target domain. For other t, we have

$$\boldsymbol{\Phi}(t) = \boldsymbol{P}_{\mathcal{S}}\boldsymbol{U}_1\boldsymbol{\Gamma}(t) - \boldsymbol{R}_{\mathcal{S}}\boldsymbol{U}_2\boldsymbol{\Sigma}(t), \tag{4.1}$$

where $\boldsymbol{U}_1 \in \mathbb{R}^{\mathsf{d}\times\mathsf{d}}$ and $\boldsymbol{U}_2 \in \mathbb{R}^{(\mathsf{D}-\mathsf{d})\times\mathsf{d}}$ are orthonormal matrices. They are given by the following pair of SVDs,

$$\boldsymbol{P}_{\mathcal{S}}^{\mathrm{T}}\boldsymbol{P}_{\mathcal{T}} = \boldsymbol{U}_1\boldsymbol{\Gamma}\boldsymbol{V}^{\mathrm{T}}, \quad \boldsymbol{R}_{\mathcal{S}}^{\mathrm{T}}\boldsymbol{P}_{\mathcal{T}} = -\boldsymbol{U}_2\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}\,. \tag{4.2}$$

$\boldsymbol{\Gamma}$ and $\boldsymbol{\Sigma}$ are $\mathsf{d}\times\mathsf{d}$ diagonal matrices. The diagonal elements are $\cos\theta_i$ and $\sin\theta_i$ for $i = 1, 2, \ldots, \mathsf{d}$. In particular, θ_i are called the principal angles between $\boldsymbol{P}_{\mathcal{S}}$ and $\boldsymbol{P}_{\mathcal{T}}$:

$$0 \leq \theta_1 \leq \theta_2 \leq \cdots \leq \theta_{\mathsf{d}} \leq \pi/2. \tag{4.3}$$

They measure the degree that subspaces "overlap". We provide the details of how to efficiently compute them in Chapter 6. Moreover, $\boldsymbol{\Gamma}(t)$ and $\boldsymbol{\Sigma}(t)$ are diagonal matrices whose elements are $\cos(t\theta_i)$ and $\sin(t\theta_i)$ respectively.

4.3.2 Compute the geodesic flow kernel (GFK)

The geodesic flow parameterizes how the source domain smoothly changes to the target domain. Consider the subspace $\boldsymbol{\Phi}(t)$ for a $t \in (0,1)$ and compute $\boldsymbol{\Phi}(t)^{\mathrm{T}}\boldsymbol{x}$, ie, the projection of a feature

[1] A similar idea was pursued in (Gopalan et al., 2011). We contrast it to our work in Chapter 9.

vector $\boldsymbol{x}$ into this subspace. If $\boldsymbol{x}$ is from the source domain and t is close to 1, then the projection will appear as if it is more likely coming from the target domain, and conversely for t close to 0. Thus, using the projection to build a classifier would result in a model using a set of features that are characteristic of both domains. Hence, this classifier would likely perform well on the target domain.

Which (or which set of) t should we use then? Our answer is surprising at the first glance: *all of them*! Intuitively, by expanding the original features with projections into **all** subspaces, we force a measurement of similarity (as we will be using inner products to construct classifiers) that is robust to any variation that leans either toward the source or towards the target or in between. In other words, the net effect is a representation that is insensitive to idiosyncrasies in either domain. We provide more detailed analysis and empirical evidence in section 4.5.

Computationally, however, we cannot use this representation explicitly. Nevertheless, we next show that there is no need to actually compute, store, and manipulate infinitely many projections.

For two original D-dimensional feature vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, we compute their projections into $\boldsymbol{\Phi}(t)$ for a continuous t from 0 to 1 and concatenate all the projections into infinite-dimensional feature vectors $\boldsymbol{z}_i^\infty$ and $\boldsymbol{z}_j^\infty$. The inner product between them defines our geodesic-flow kernel,

$$\langle \boldsymbol{z}_i^\infty, \boldsymbol{z}_j^\infty \rangle = \int_0^1 (\boldsymbol{\Phi}(t)^\mathsf{T} \boldsymbol{x}_i)^\mathsf{T} (\boldsymbol{\Phi}(t)^\mathsf{T} \boldsymbol{x}_j)\, dt = \boldsymbol{x}_i^\mathsf{T} \boldsymbol{G} \boldsymbol{x}_j, \tag{4.4}$$

where $\boldsymbol{G} \in \mathbb{R}^{\mathsf{D}\times\mathsf{D}}$ is a positive semidefinite matrix. This is precisely the "kernel trick", where a kernel function induces inner products between infinite-dimensional features.

The matrix $\boldsymbol{G}$ can be computed in a closed-form from previously defined matrices:

$$\boldsymbol{G} = [\boldsymbol{P}_\mathcal{S}\boldsymbol{U}_1 \;\; \boldsymbol{R}_\mathcal{S}\boldsymbol{U}_2] \begin{bmatrix} \boldsymbol{\Lambda}_1 & \boldsymbol{\Lambda}_2 \\ \boldsymbol{\Lambda}_2 & \boldsymbol{\Lambda}_3 \end{bmatrix} \begin{bmatrix} \boldsymbol{U}_1^\mathsf{T} \boldsymbol{P}_\mathcal{S}^\mathsf{T} \\ \boldsymbol{U}_2^\mathsf{T} \boldsymbol{R}_\mathcal{S}^\mathsf{T} \end{bmatrix} \tag{4.5}$$

where $\boldsymbol{\Lambda}_1$ to $\boldsymbol{\Lambda}_3$ are diagonal matrices, whose diagonal elements are

$$\lambda_{1i} = 1 + \frac{\sin(2\theta_i)}{2\theta_i}, \lambda_{2i} = \frac{\cos(2\theta_i) - 1}{2\theta_i}, \lambda_{3i} = 1 - \frac{\sin(2\theta_i)}{2\theta_i}. \tag{4.6}$$

Detailed derivations are given in Appendix A.

4.3.3 Extract the domain-invariant feature space

The kernel $\boldsymbol{G}$ can be plugged into any kernelized classifiers, such as nonlinear SVMs. Additionally, we can also extract from it an equivalent finite-dimensional domain-invariant feature space. Let $\boldsymbol{L}$ be $\boldsymbol{G}$'s square root: $\boldsymbol{L}^\mathsf{T}\boldsymbol{L} = \boldsymbol{G}$. The domain-invariant feature space is given by the following feature mapping

$$\boldsymbol{x} \to \boldsymbol{z} = \boldsymbol{L}\boldsymbol{x}, \tag{4.7}$$

such that $\boldsymbol{z}_i^\mathsf{T}\boldsymbol{z}_j = \boldsymbol{x}_i^\mathsf{T}\boldsymbol{G}\boldsymbol{x}_j$. This explicit feature representation is convenient for constructing other types of classifiers that do not depend on inner products (such as logistic regression).

The closed-form expression of the geodesic flow kernel is convenient to use and does not depend on user-selected parameters such as the bandwidth in the Gaussian RBF kernels. In practice, we need to choose the dimensionality d of the subspaces for representing the domains.

We will show next how to automatically infer this hyperparameter from the data, thus making the proposed method fully automatic and free of tuning any hyperparameters.

4.4 Automatic inference of subspace dimension d

The intuition behind our approach of automatically inferring the dimensionality d of the subspaces is to align as much as possible the subspaces of the source and target domains. To this end, we develop a subspace disagreement metric (SDM).

To compute SDM, we first compute the PCA subspaces of the two datasets, $\mathbf{PCA}_{\mathcal{S}}$ and $\mathbf{PCA}_{\mathcal{T}}$. We also combine the datasets into one dataset and compute its subspace $\mathbf{PCA}_{\mathcal{S}+\mathcal{T}}$. Intuitively, if the two datasets are similar, then all three subspaces should not be too far away from each other on the Grassmannian. The SDM captures this notion and is defined in terms of the principal angles (cf. eq. (4.3)),

$$\mathcal{D}(\mathsf{d}) = 0.5 \left[\sin \alpha_{\mathsf{d}} + \sin \beta_{\mathsf{d}}\right], \tag{4.8}$$

where α_{d} denotes the d-th principal angle between the $\mathbf{PCA}_{\mathcal{S}}$ and $\mathbf{PCA}_{\mathcal{S}+\mathcal{T}}$ and β_{d} between $\mathbf{PCA}_{\mathcal{T}}$ and $\mathbf{PCA}_{\mathcal{S}+\mathcal{T}}$. The quantity $\sin \alpha_{\mathsf{d}}$ or $\sin \beta_{\mathsf{d}}$ is called the minimum correlation distance (Hamm & Lee, 2008).

Note that $\mathcal{D}(\mathsf{d})$ is at most 1. A small value indicates that both α_{d} and β_{d} are small, thus $\mathbf{PCA}_{\mathcal{S}}$ and $\mathbf{PCA}_{\mathcal{T}}$ are aligned (at the d-th dimension). At its maximum value of 1, the two subspaces have orthogonal directions (i.e., $\alpha_d = \beta_d = \pi/2$). In this case, domain adaptation will become difficult as variances captured in one subspace would not be able to transfer to the other subspace.

To identify the optimal d, we adopt a greedy strategy:

$$\mathsf{d}^* = \min\{\mathsf{d} | \mathcal{D}(\mathsf{d}) = 1\}. \tag{4.9}$$

Intuitively, the optimal d^* should be as high as possible (to preserve variances in the source domain for the purpose of building good classifiers) but should not be so high that the two subspaces start to have orthogonal directions.

4.5 Reducing domain discrepancy with the GFK

Building on the general intuitions described above, we now more formally justify the rationale behind the GFK. How exactly does GFK reduce the discrepancy across domains? The definition of the kernel in eq. (4.4) provides several clues. In particular, we will show in the following that the proposed kernel construction leads to measuring distances between data points in a way that is insensitive to domains.

To start with, consider that we would like to use a nearest neighbor classifier on both the source and the target domains in an ideal domain-invariant feature subspace $\mathcal{F}$, parameterized by its basis $\boldsymbol{F}$. What properties do we desire for the subspace $\mathcal{F}$? In what follows, we describe two such properties which are strongly correlated with empirical evidence in supporting using the GFK for deriving domain-invariant features.

Table 4.1: Distortion ratios (in %) to distances computed within the source and target domains, using 4 subspaces

Domain pairs	PCA$_{\mathcal{S}}$	PCA$_{\mathcal{T}}$	PCA$_{\mathcal{S}+\mathcal{T}}$	GFK
AMAZON- CALTECH	8.78	7.71	5.48	6.18
AMAZON- DSLR	19.9	17.3	15.9	13.2
AMAZON- WEBCAM	15.5	14.0	11.8	10.8
CALTECH- DSLR	14.1	16.3	12.1	11.1
CALTECH- WEBCAM	15.5	14.8	11.0	10.9
DSLR- WEBCAM	15.7	13.7	10.4	10.6
Average	14.9	14.0	11.1	**10.5**

At the foremost, we would like $\mathcal{F}$ to preserve distances between data points measured within the source domain's subspace. Namely,

$$\|\boldsymbol{F}^{\mathrm{T}}\boldsymbol{x}_i - \boldsymbol{F}^{\mathrm{T}}\boldsymbol{x}_j\|_2^2 - \|\boldsymbol{P}_{\mathcal{S}}^{\mathrm{T}}\boldsymbol{x}_i - \boldsymbol{P}_{\mathcal{S}}^{\mathrm{T}}\boldsymbol{x}_j\|_2^2 \approx 0, \tag{4.10}$$

for points $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{S}$. This is analogous to the central ideas of many manifold learning algorithms to preserve distances. Similarly, for a pair of data points $\boldsymbol{x}_m$ and $\boldsymbol{x}_n$ from the target domain $\mathcal{T}$, we would like

$$\|\boldsymbol{F}^{\mathrm{T}}\boldsymbol{x}_m - \boldsymbol{F}^{\mathrm{T}}\boldsymbol{x}_n\|_2^2 - \|\boldsymbol{P}_{\mathcal{T}}^{\mathrm{T}}\boldsymbol{x}_m - \boldsymbol{P}_{\mathcal{T}}^{\mathrm{T}}\boldsymbol{x}_n\|_2^2 \approx 0. \tag{4.11}$$

Along the geodesic flow eq. (4.1), if we select the subspace $\boldsymbol{F} = \boldsymbol{\Phi}(\boldsymbol{t})$ with $t \ll 1$, the distance-preserving condition for the source domain is easy to be satisfied as $\boldsymbol{F}$ would be close to $\boldsymbol{P}_{\mathcal{S}}$. However, such $\boldsymbol{F}$ would distort the distance-preserving condition for the target domain significantly as $\boldsymbol{F}$ would be very different from $\boldsymbol{P}_{\mathcal{T}}$. Conversely, for $t \approx 1$, the selected subspace $\boldsymbol{F}$ on the flow will preserve distances in the opposite way.

The "right" choice is then to balance the averaged distortion ratios (i.e., distortions divided by distances) for each domain and select an intermediate point on the flow. While feasible in theory (for instance, by minimizing a properly constructed objective function over t), an alternative approach is to use *all subspaces*, as in the derivation of our GFK. The intuition is "to average out": for any t, if the subspace $\boldsymbol{\Phi}(t)$ preserves the distances for the source domain better than the target domain, then for $(1-t)$, the converse is true. In other words, if uniformly sampling the flow, the expected distortion is the same for both domains — we are not favoring any particular one of them. More precisely, this subspace will give rise to the following distance function

$$\|\boldsymbol{F}^{\mathrm{T}}\boldsymbol{x} - \boldsymbol{F}^{\mathrm{T}}\boldsymbol{x}'\|_2^2 = (\boldsymbol{x} - \boldsymbol{x}')^{\mathrm{T}} \int_t \boldsymbol{\Phi}(t)^{\mathrm{T}}\boldsymbol{\Phi}(t)dt\,(\boldsymbol{x} - \boldsymbol{x}'), \tag{4.12}$$

which is precisely defined in terms of our GFK.

To help illustrate this point, Table 4.1 reports the averaged distortion ratios computed within the source and target domains, using four different subspaces: the PCA subspace of the source (PCA$_{\mathcal{S}}$), the PCA subspace of the target (PCA$_{\mathcal{T}}$), the PCA subspace of merging the source and the target (PCA$_{\mathcal{S}+\mathcal{T}}$), and the subspace induced by our GFK (GFK). We report results on six different pairs of source and target domains, all used in our experimental studies for domain adaptation in Chapter 9. The subspace by our GFK attains the smallest distortion.

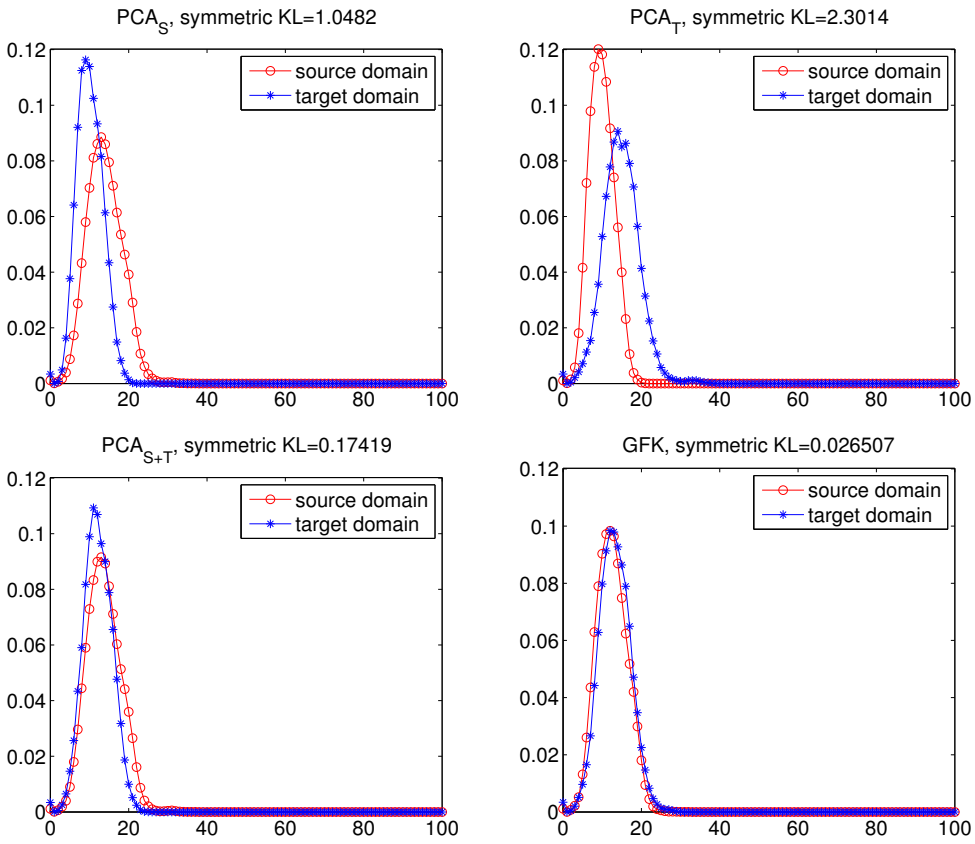

Figure 4.2: Histograms of pairwise distances within each domain where the distances are calculated within four different subspaces. GFK induces a subspace such that the difference between the source's histogram and the target's is the smallest.

The second property we desire for $\mathcal{F}$ is closely related to our goal of using the source labeled data to classify the target unlabeled data. Intuitively, if we use $\boldsymbol{F}$ to measure pairwise distances within each domain, then those two sets of distances should be similarly distributed. Otherwise, the data instances in the target domain will not be classified as effectively as the instances in the source domain. This is especially true if data follows the assumption of discriminative clustering — all instances from the same class form a tight cluster and different clusters tend to be apart from each other (Shi & Sha, 2012).

Fig. 4.2 displays the histograms of those pairwise distances computed using several subspaces. The source domain is the AMAZON dataset and the target domain is the WEBCAM dataset (for a detailed description of these datasets, please refer to section 8.5). We see that the subspace corresponding to GFK brings the source and the target domains closest. Table 4.2 quantitatively confirms the outcome; we report the symmetric KL divergence between those histograms for the same six pairs of source and target domains as in Table 4.1. Clearly, GFK is able to attain the smallest divergences.

Combining the results in Table 4.1 and 4.2, we find that the GFK leads to a subspace that best satisfies the two desirable properties simultaneously: minimal distortions to distances and

Table 4.2: Symmetric KL divergences between the histograms of pairwise distances across two domains

Domain pairs	PCA$_{\mathcal{S}}$	PCA$_{\mathcal{T}}$	PCA$_{\mathcal{S}+\mathcal{T}}$	GFK
AMAZON- CALTECH	0.413	0.445	0.014	0.012
AMAZON- DSLR	2.145	7.411	0.734	0.33
AMAZON- WEBCAM	1.048	2.301	0.174	0.027
CALTECH- DSLR	1.026	2.488	0.587	0.138
CALTECH- WEBCAM	1.747	2.188	0.347	0.178
DSLR- WEBCAM	2.884	0.808	0.009	0.089
Average	1.544	2.607	0.311	**0.129**

matching how distances are distributed. This empirical evidence strongly supports the GFK as a method to extract domain-invariant features. This support is echoed by the superior performance of GFK in benchmark problems, reported in Chapter 9.

4.6 Summary

To recap our GFK-based approach, we i) determine the optimal dimensionality of the subspaces (eq. (4.9)); ii) compute the geodesic flow kernel $\boldsymbol{G}$ using the subspaces eq. (4.5); iii) use the kernel to construct a classifier with the labeled data, either using a kernelized classifier which requires only the inner products defined by the kernel matrix $\boldsymbol{G}$ or using the invariant features in eq. (4.7) in other classifiers.

Chapter 5

Landmarks: A New Intrinsic Structure for Domain Adaptation

In the GFK approach described so far—as well as more generally in existing domain adaptation work that models the mismatch and similarity between the source and target domains—all data instances in the source domain are treated "holistically", with the assumption that all of them are equally adaptable to the target domain. In contrast, here our insight is that instead modeling adaptation at a finer granularity will be beneficial. In particular, we hypothesize that not all instances from the source domain are equally amenable to adaptation.

As a motivating example, suppose the source domain contains furniture in a home environment and the target domain consists of images of office-style furniture. Conceivably, certain images from the source — such as those taken in home offices — could also be regarded as samples from the target domain. Such images thus might have properties that are shared by both domains. These properties in turn can guide learning algorithms to search for invariant features.

Our approach automatically discovers and identifies such images, which we call "landmarks". We use them to bridge the source and the target domains to generate multiple candidates of invariant feature spaces. Additionally, we exploit the labels of the landmarks to adapt discriminatively the invariant features to be optimal for the target domain.

In contrast to the GFK approach discussed thus far, our landmark-based approach discovers feature spaces that are *discriminatively* optimized. On the other hand, we also show that the landmark-based approach integrates well with GFK. In particular, our automatic landmark identification algorithm (section 5.2) benefits significantly from using the GFK as a similarity measure.

We start by giving an overview of the landmark approach, followed by details on how to identify landmarks. We then show how to exploit those landmarks for discriminative learning on the target domain.

5.1 Main idea

As the first step, our landmark approach plucks out and exploits the most desirable instances — landmarks — to facilitate adaptation. Identifying those instances requires comparing all possible subsets from the source domain to the target. We will show how this can be addressed with tractable optimization.

Leveraging the existence of landmarks and their properties, we create a cohort of auxiliary tasks where landmarks explicitly bridge the source and target domains. Specifically, in those auxiliary tasks, the original target domain is augmented with landmarks, blurring the distinction

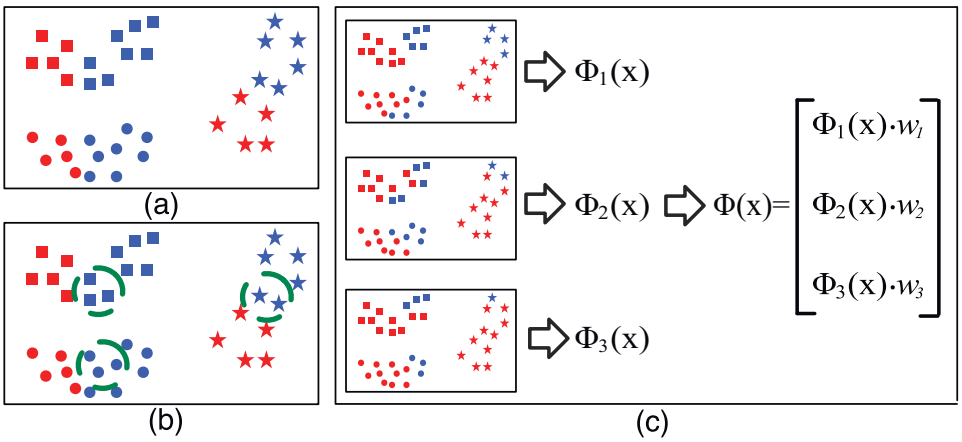

Figure 5.1: Sketch of the main idea of our landmark approach (best viewed in color). (a) The original domain adaptation (DA) problem where instances in red are from the target and in blue from the source. (b) **Landmarks**, shown inside the green circles, are data instances from the source that can be regarded as samples from the target (section 5.2). (c) Multiple auxiliary tasks are created by augmenting the original target with landmarks, which switches their color (domain association) from blue to red (section 5.3). Each task gives rise to a new feature representation. These representations are combined discriminatively to form domain-invariant features for the original DA problem (section 5.4).

across domains. Thus, those tasks are *easier* to solve than the original problem. We show this is indeed true both theoretically and empirically.

The auxiliary tasks offer multiple views of the original problem. In particular, each task differs by how its landmarks are selected, which in turn is determined by how the similarity among instances is measured. In this work, we measure similarities at multiple scales (of distances). Thus, each view provides a different perspective on the adaptation problem by being robust to idiosyncrasies in the domains at different granularities.

The solutions of the auxiliary tasks give rise to multiple domain-invariant feature spaces that can be characterized by linear positive semidefinite kernel functions. We parameterize invariant features for the original adaptation problem with those auxiliary kernels. Intuitively, not all of the kernels are equally useful; to discern which are, we cast the corresponding learning problem in terms of multiple kernel learning. We learn the kernel discriminatively to minimize classification errors on the landmark data instances, which serve as a proxy to discriminative loss on the target domain. Fig. 5.1 schematically illustrates the overall approach.

We describe our three-step landmark-based approach below: i) identifying and selecting the landmark instances; ii) constructing multiple auxiliary tasks using landmarks and inferring the corresponding domain-invariant feature spaces, one for each auxiliary task; iii) discriminatively learning the final domain-invariant feature space that is optimized for the target domain.

5.2 Discovering landmarks

Landmarks are data points from the source domain; however, given how they are distributed, they look like they could be samples from the target domain too (*cf.* Fig. 5.1 for a schematic illustration, and Fig. 9.3 in Chapter 8.5 for exemplar images of visual objects identified as landmarks in vision datasets). The intuition behind our approach is to use these landmarks to bridge the source and the target domains.

How can we identify those landmarks? At first glance, it seems that we need to compare all possible subsets of training instances in the source domain to the target. We will show in the following this seemingly intractable problem can be relaxed and solved with tractable convex optimization.

Let $\mathcal{S} = \{(\boldsymbol{x}_m, y_m)\}_{m=1}^{\mathsf{M}}$ denote M data points and their labels from the source domain. Likewise, we use $\mathcal{T} = \{\boldsymbol{x}_n\}_{n=1}^{\mathsf{N}}$ for the target domain.

5.2.1 Landmark selection

To identify landmarks, we use M indicator variables $\boldsymbol{\alpha} = \{\alpha_m \in \{0, 1\}\}$, one for each data point in the source domain. If $\alpha_m = 1$, then $\boldsymbol{x}_m$ is regarded as a landmark. Our goal is to choose among all possible configurations of $\boldsymbol{\alpha} = \{\alpha_m\}$ such that the distribution of the *selected* data instances is maximally similar to that of the target domain.

To determine whether the two distributions are similar, we use a non-parametric two-sample test called maximum mean discrepancy (MMD) (Gretton et al., 2006) (other approaches are also possible, including building density estimators when the dimensionality is not high). Specifically, we use a nonlinear feature mapping function $\phi(\cdot)$ to map $\boldsymbol{x}$ to a Reproducing Kernel Hilbert Space (RKHS) and compare the difference in sample means. When the mapping function is a unit-ball in a universal RKHS, the difference can be conveniently calculated in the following[1],

$$\mathsf{MMD}(\boldsymbol{\alpha}) = \left\| \frac{1}{\sum_m \alpha_m} \sum_m \alpha_m \phi(\boldsymbol{x}_m) - \frac{1}{\mathsf{N}} \sum_n \phi(\boldsymbol{x}_n) \right\|_{\mathcal{H}}^2, \tag{5.1}$$

where $\sum \alpha_m$ is the number of selected landmarks, and the first term inside the norm is the mean of the selected landmarks under the mapping.

Our goal is to choose $\boldsymbol{\alpha}$ such that the difference is minimized. Furthermore, we impose the constraint that labels be *balanced* in the selected landmarks. Concretely, we arrive at the following optimization problem

$$\min_{\boldsymbol{\alpha}} \quad \mathsf{MMD}(\boldsymbol{\alpha}) \tag{5.2}$$

$$\text{s.t.} \quad \frac{1}{\sum_m \alpha_m} \sum_m \alpha_m y_{mc} = \frac{1}{\mathsf{M}} \sum_m y_{mc}, \tag{5.3}$$

[1]The unit-ball condition allows the difference be represented as a metric in the form of eq. (5.1) and the universality ensures that the means are injective such that the difference in the means is zero if and only if the two distributions are the same. For more detailed theoretical analysis, please refer to (Gretton et al., 2006).

where y_{mc} is an indicator variable for $y_m = c$. The right-hand-side of the constraint is simply the prior probability of the class c, estimated from the source.

We stress that the above criterion is defined on landmarks, which are a *subset* of the source domain, as the sample mean is computed *only* on the selected instances (*cf.* the denominator $\sum_m \alpha_m$ in eq. (5.2)). This is very different from other approaches that have used similar non-parametric techniques for comparing distributions (Pan et al., 2009; Gretton et al., 2009). There they make stronger assumptions that all data points in the source domain need to be collectively distributed similarly to the target domain. Furthermore, they do not impose the balance constraint of eq. (5.3). Our results will show that these differences are crucial to the success of our approach.

Eq. (5.2) is intractable due to the binary constraints on α_m. We relax and solve it efficiently with convex optimization. We define new variables β_m as $\alpha_m \left(\sum_m \alpha_m\right)^{-1}$. We relax them to live on the simplex $\Delta = \{\boldsymbol{\beta} : \beta_m \geq 0, \sum_m \beta_m = 1\}$. Substituting $\{\beta_m\}$ into eq. (5.2) and its constraints, we arrive at the following quadratic programming problem:

$$\begin{array}{rl} \min_{\boldsymbol{\beta}\in\Delta} & \boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{\beta} - 2/\mathsf{N}\,\boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{B}\mathbf{1} \\ \text{s.t.} & \sum_m \beta_m y_{mc} = 1/\mathsf{M}\sum_m y_{mc}, \quad \forall\, c, \end{array} \tag{5.4}$$

where $\boldsymbol{A} \in \mathbb{R}^{\mathsf{M}\times\mathsf{M}}$ denotes the kernel matrix computed over the source domain, and $\boldsymbol{B} \in \mathbb{R}^{\mathsf{M}\times\mathsf{N}}$ denotes the kernel matrix computed between the source domain points and target domain points. The optimization is convex, as the kernel matrix $\boldsymbol{A}$ is positive semidefinite.

We recover the binary solution for α_m by finding the support of β_m, ie, $\alpha_m = \text{THRESHOLD}(\beta_m)$. In practice, we often obtain *sparse* solutions, supporting our modeling intuition that only a subset of instances in the source domain is needed to match the target domain.

5.2.2 Multi-scale analysis

The selection of landmarks depends on the kernel mapping $\phi(\boldsymbol{x})$ and its parameter(s). To satisfy the requirement of being a unit-ball in a universal RKHS, we use Gaussian RBF kernels, defined as follows:

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\{-(\boldsymbol{x}_i - \boldsymbol{x}_j)^{\mathsf{T}}\boldsymbol{M}(\boldsymbol{x}_i - \boldsymbol{x}_j)/\sigma^2\}, \tag{5.5}$$

where the metric $\boldsymbol{M}$ is positive semidefinite. We experimented with several choices — details in Chapter 9.

The bandwidth σ is a scaling factor for measuring distances and similarities between data points. Since we regard landmarks as likely samples from the target domain, σ determines how much the source and the target are similar to each other at different granularities. A small σ will attenuate distances rapidly and regard even close points as being dissimilar. Thus, it is likely to select a *large* number of points as landmarks in order to match distributions. A large σ will have the opposite effect. Fig. 9.3 illustrates the effect of σ.

Instead of choosing one σ in the hope that one scale fits all, we devise a multiscale approach. We use a set $\{\sigma_q \in [\sigma_{min},\ \sigma_{max}]\}_{q=1}^{\mathsf{Q}}$. For each σ_q, we compute the kernel according to eq. (5.5) and solve eq. (5.4) to obtain the corresponding landmarks $\mathcal{L}^q = \{(\boldsymbol{x}_m, y_m) : \alpha_m = 1\}$. Using multiple scales adds the flexibility of modeling data where similarities cannot be measured in one homogeneous scale. For example, the category of GRIZZLY BEAR is conceivably much closer to GREY BEAR than to POLAR BEAR, and so to capture similarities among both the pairs as well as among all three, it is necessary to model them at two scales.

Each set of landmarks (one set per scale) gives rise to a different perspective on the adaptation problem by suggesting which instances to explore to connect the source and the target. We achieve this connection by creating auxiliary tasks, as we describe next.

5.3 Constructing auxiliary tasks

Imagine we create a new source domain $\mathcal{S}^q = \mathcal{S} \setminus \mathcal{L}^q$ and a new target domain $\mathcal{T}^q = \mathcal{T} \bigcup \mathcal{L}^q$, where the $\mathcal{L}^q$ is removed from and added to the source and target domains, respectively. We do not use $\mathcal{L}^q$'s labels at this stage yet.

Our auxiliary tasks are defined as Q domain adaptation problems, $\mathcal{S}^q \rightarrow \mathcal{T}^q$. The auxiliary tasks differ from the original problem $\mathcal{S} \rightarrow \mathcal{T}$ in an important aspect: the new tasks should be "easier", as the existence of landmark points ought to aid the adaptation. This is illustrated by the following theorem, stating that the discrepancy between the new domains is smaller than the original.

Let $P_S(X)$ and $P_T(X)$ denote the distributions of the original source and the target domains, respectively. Suppose $P_S(X) = \alpha P_N(X) + (1-\alpha)P_L(X)$ with $\alpha \in [0, 1)$ is a mixture model where $P_L(X)$ is the component corresponding to the landmark data points and $P_N(X)$ corresponds to the distribution of the non-landmark instances. For the auxiliary task, assume the new target distribution is modeled as a mixture distribution $Q_T(X) = \beta P_T(X) + (1-\beta)P_L(X)$ where $\beta \in [0,\ 1)$. Furthermore, assume the source distribution remains essentially unchanged, which is easily satisfied as long as the number of instances in the source domain is significantly greater than the number of landmark instances and the landmarks are selected *i.i.d.* from $P_L(X)$[2].

In what follows, we omit the arguments and their parentheses (X) to simplify the notation.

Theorem 4. *The following inequality holds,*

$$KL(P_S \| Q_T) \leq KL(P_S \| P_T)$$

where $KL(\cdot\|\cdot)$ stands for the Kullback-Leibler divergence, if the following condition is satisfied

$$\begin{aligned} &\alpha KL(P_N \| P_T) + (1-\alpha) KL(P_L \| P_T) \\ &\qquad \geq \frac{9}{8} \max\left\{KL(P_L \| P_N),\ KL(P_N \| P_L)\right\} \end{aligned} \tag{5.6}$$

In words, the new target distribution is closer to the source distribution, on the condition that the inter-domain difference (i.e. the left-hand-side) is greater than the intra-domain discrepancy or inhomogeneity (i.e., the right-hand-side).

The proof is in the Appendix B. Note that the condition in eq. (5.6) is mild: we would expect the source domain is relatively homogeneous and is distinct from the target domain. Also note that the theorem subsumes a similar result in our previous work (Gong et al., 2013b) where we have assumed that the landmarks are *i.i.d* sampled from the source domain, which is a more stringent assumption that corresponds to the current theorem when $P_L(X) = P_N(X)$.

[2]Note that we do not require the landmarks to be *i.i.d* samples from $P_S(X)$ — they only need to be representative samples of $P_L(X)$.

With the reduced discrepancy between $P_S(X)$ and $Q_T(X)$, we can apply the analysis in (Mansour et al., 2009a, Lemma 1) to show that classifiers applied to $Q_T(X)$ attain a smaller generalization error bound than those applied to $P_T(X)$. Intuitively, the increased similarity between the new domains is also closely related to the increased difficulty of distinguishing which domain a data point is sampled from. More formally, if we were to build a binary classifier to classify a data point into one of the two categories SOURCE versus TARGET, we would expect the accuracy to drop when we compare the original to the auxiliary tasks. The accuracy — also named as *A-distance* — is closely related to how effective domain adaption can be (Blitzer et al., 2007). A high accuracy is indicative of a highly contrasting pair of domains, and thus is possibly due to many domain-specific features capturing each domain's individual characteristics.

These insights motivate our design of auxiliary tasks: they conceivably have low accuracy for binary classification as the landmarks blend the two domains, discouraging the use of domain-specific features. We describe next how to extract domain-invariant ones using the solutions of those easy problems as a *basis*.

5.3.1 Learning basis from auxiliary tasks

Having shown that the auxiliary tasks represent easier domain adaptation problems, we now use them for adaptation. Specifically, for every pair of auxiliary domains, we use the geodesic flow kernel to compute domain-invariant features. The GFK is particularly adept at measuring domain-invariant distances among data points, as exemplified by its superior performance in nearest-neighbor classifiers (*cf.* Section 9.2). Thus, it is especially suitable for the final stage of our approach when we compose complex domain-invariant features (*cf.* Section 5.4).

The domain-invariant feature space is extracted as the mapping $\Phi_q(\boldsymbol{x}) = \boldsymbol{L}_q\boldsymbol{x}$, where $\boldsymbol{G}_q = \boldsymbol{L}_q^{\mathsf{T}}\boldsymbol{L}_q$ is the GFK for the q-th auxiliary task (cf. eq. (4.7)). In the following, we describe how to integrate the spaces — one for each auxiliary task — *discriminatively* so that the final feature space is optimal for the target.

5.4 Discriminative learning

In this final step, we reveal the second use of landmarks beyond constructing auxiliary tasks. We will use their labels to learn *discriminative* domain-invariant features for the target domain. Concretely, we compose the features for the original adaptation problem with the auxiliary tasks' features as a basis.

We scale and concatenate those features $\{\sqrt{w_q}\Phi_q(\boldsymbol{x})\}_{q=1}^{\mathsf{Q}}$ into a super-feature vector $\boldsymbol{f}$. Learning $\{w_q\}$ is cast as learning a convex combination of all kernels $\boldsymbol{G}_q$ (Lanckriet et al., 2004),

$$\boldsymbol{F} = \sum_q w_q \boldsymbol{G}_q, \text{ s.t. } \; w_q \geq 0 \text{ and } \sum_q w_q = 1. \tag{5.7}$$

We use the kernel $\boldsymbol{F}$ in training a SVM classifier and the labels of the landmarks $\{\mathcal{L}^q\}$, i.e., $\mathcal{D}_{\text{TRAIN}} = \sum_q \mathcal{L}^q$ to optimize $\{w_q\}$ discriminatively. We use $\mathcal{D}_{\text{DEV}} = \mathcal{S} \setminus \mathcal{D}_{\text{TRAIN}}$ be a validation dataset for model selection. Since $\mathcal{D}_{\text{TRAIN}}$ consists of landmarks that are distributed similarly to the target, we expect the classification error on $\mathcal{D}_{\text{TRAIN}}$ to be a good proxy to that of the target.

5.5 Summary

To recap our landmark-based approach: i) at each granularity σ_q, and with the aid of our GFK, we automatically select *landmarks* — individual instances that are distributed most similarly to the target; ii) we then construct *auxiliary* tasks and use their solutions as a basis for composing domain-invariant features; iii) we learn features *discriminatively*, using classification loss on the landmarks as a proxy to the discriminative loss on the target.

Chapter 6

Rank of Domains

Imagine we need build a classifier for a target domain for object recognition. We have several datasets, Caltech-101, PASCAL VOC, and ImageNet to choose from as the source domain[1]. Without actually running our domain adaptation algorithms and building classifiers, is it possible to determine which dataset(s) would give us the best performance on the target domain? This question is of practical importance: it is much more cost-effective to be able to select one (or a limited few) that is likely to adapt well to the target domain, instead of trying each one of them.

To answer this question, we introduce a Rank-of-Domains (ROD) metric that integrates two sets of information: geometrically, the alignment between subspaces, and statistically, KL divergences between data distributions once they are projected into the subspaces. The ROD metric is defined upon the concepts of principal angles and principal vectors between two subspaces.

6.1 Principal angles and principal vectors

Let $\boldsymbol{P}_{\mathcal{S}}, \boldsymbol{P}_{\mathcal{T}} \in \mathbb{R}^{\mathsf{D}\times\mathsf{d}}$ be the bases of the source subspace and the target subspace, respectively. The *principal angles* $\{\theta_i\}$ between the two subspaces are recursively defined as,

$$\cos(\theta_i) = \max_{\boldsymbol{s}_i \in \mathrm{span}(\boldsymbol{P}_{\mathcal{S}})} \max_{\boldsymbol{t}_i \in \mathrm{span}(\boldsymbol{P}_{\mathcal{T}})} \frac{\langle \boldsymbol{s}_i, \boldsymbol{t}_i \rangle}{\|\boldsymbol{s}_i\| \|\boldsymbol{t}_i\|}, \quad , i = 1, 2, \cdots, \mathsf{d} \tag{6.1}$$

such that

$$\begin{array}{ll} \boldsymbol{s}_k \in \mathrm{span}(\boldsymbol{P}_{\mathcal{S}}), & \boldsymbol{s}_i \perp \boldsymbol{s}_k, \\ \boldsymbol{t}_k \in \mathrm{span}(\boldsymbol{P}_{\mathcal{T}}), & \boldsymbol{t}_i \perp \boldsymbol{t}_k, \end{array} \; k = 1, 2, \cdots i-1.$$

In the above, $\boldsymbol{s}_i$ and $\boldsymbol{t}_i$ are called the *principal vectors* associated with θ_i, and $\mathrm{span}(\boldsymbol{P})$ denotes the subspace spanned by the column vectors of $\boldsymbol{P}$. Essentially, the principal vectors $\{\boldsymbol{s}_i\}$ and $\{\boldsymbol{t}_i\}$ are the new bases for the two subspaces, respectively, such that after the change of the bases, the two subspaces maximally overlap along the "ordered" bases. The degrees of overlapping are measured by the principal angles — the series of smallest angles between the bases.

Now we are ready to define the ROD metric.

[1] In addition to picking out a single best source domain, of course multi-source domain adaptation is another choice. Nonetheless, the computation cost is often higher. It also bears a higher risk of negative adaptation if some of the source domains are dramatically different from the target domain. In other words, multi-source domain adaptation does not necessarily outperform adapting from a single good source domain.

6.2 Rank of Domains (ROD)

Given a pair of domains, ROD involves three steps: i) determine the optimal dimensionality d^* for the subspaces (as in section 4.4); ii) at each dimension $i \leq \mathsf{d}^*$, approximate the data distributions of the two domains with two one-dimensional Gaussians and then compute the symmetrized KL divergences between them; iii) compute the KL-divergence weighted average of principal angles,

$$\mathcal{R}(\mathcal{S}, \mathcal{T}) = \frac{1}{\mathsf{d}^*} \sum_{i}^{\mathsf{d}^*} \theta_i \left[KL(\mathcal{S}_i \| \mathcal{T}_i) + KL(\mathcal{T}_i \| \mathcal{S}_i) \right]. \tag{6.2}$$

$\mathcal{S}_i$ and $\mathcal{T}_i$ are the two above-mentioned Gaussian distributions; they are estimated from data projected onto the principal vectors (associated with the i-th principal angle). Note we use only the first d^* directions. Beyond that, the subspaces of the two domains start to have orthogonal directions, on which the two domains would have very different geometric and statistical properties. As such, the source classifier is unlikely to be adapted successfully to the target.

A pair of domains with smaller values of $\mathcal{R}(\mathcal{S}, \mathcal{T})$ are more likely to adapt well: the two domains are both geometrically well-aligned (small principal angles) and similarly distributed (small KL divergences). Empirically, when we use the metric to rank various datasets as source domains, we find the ranking correlates well with their relative performance on the target domain.

6.3 Computing ROD

Let $\boldsymbol{X}_{\mathcal{S}} \in \mathbb{R}^{\mathsf{M} \times \mathsf{D}}$ and $\boldsymbol{X}_{\mathcal{T}} \in \mathbb{R}^{\mathsf{N} \times \mathsf{D}}$ denote the observed input samples from the source and the target domains. We provide the details on how to compute the ROD metric between them. The corresponding subspaces $\boldsymbol{P}_{\mathcal{S}}, \boldsymbol{P}_{\mathcal{T}} \in \mathbb{R}^{\mathsf{D} \times \mathsf{d}^*}$ are obtained using standard PCA, where the subspace dimension d^* is automatically determined by the subspace disagreement measure in Section 4.4.

The next step is to calculate the principal angels and vectors. Given the singular value decomposition, $\boldsymbol{P}_{\mathcal{S}}^{\mathrm{T}} \boldsymbol{P}_{\mathcal{T}} = \boldsymbol{U}_1 \boldsymbol{\Gamma} \boldsymbol{V}^{\mathrm{T}}$, the principal angles and vectors can be solved in closed from,

$$\theta_i = \arccos \gamma_i, \boldsymbol{s}_i = (\boldsymbol{P}_{\mathcal{S}} \boldsymbol{U}_1)_{\cdot,i}, \boldsymbol{t}_i = (\boldsymbol{P}_{\mathcal{T}} \boldsymbol{V})_{\cdot,i}, \tag{6.3}$$

where γ_i is the i-th diagonal element of the diagonal matrix $\boldsymbol{\Gamma}$. $(\boldsymbol{M})_{\cdot,i}$ returns the i-th column of the matrix $\boldsymbol{M}$.

We also need fit the one-dimensional Gaussian distributions to the data projected to each principal vector. Suppose that $\boldsymbol{X}_{\mathcal{S}}$ and $\boldsymbol{X}_{\mathcal{T}}$ have zero-means. We thus need only to compute the variances in order to specify the Gaussians,

$$\sigma_{i\mathcal{S}}^2 = \frac{1}{\mathsf{M}} \boldsymbol{s}_i^{\mathrm{T}} \boldsymbol{X}_{\mathcal{S}}^{\mathrm{T}} \boldsymbol{X}_{\mathcal{S}} \boldsymbol{s}_i, \quad \sigma_{i\mathcal{T}}^2 = \frac{1}{\mathsf{N}} \boldsymbol{t}_i^{\mathrm{T}} \boldsymbol{X}_{\mathcal{T}}^{\mathrm{T}} \boldsymbol{X}_{\mathcal{T}} \boldsymbol{t}_i. \tag{6.4}$$

Now we are ready to give the formulation to compute the ROD metric,

$$\mathcal{R}(\mathcal{S}, \mathcal{T}) = \frac{1}{\mathsf{d}^*} \sum_{i=1}^{\mathsf{d}^*} \theta_i \left[\frac{1}{2} \frac{\sigma_{i\mathcal{S}}^2}{\sigma_{i\mathcal{T}}^2} + \frac{1}{2} \frac{\sigma_{i\mathcal{T}}^2}{\sigma_{i\mathcal{S}}^2} - 1 \right]. \tag{6.5}$$

thanks to that the KL-divergence between two Gaussian distributions has an analytical solution.

Chapter 7

Discovering Latent Domains

So far we have been focusing on how to *adapt* a classifier from the labeled source domain such that it achieves good performance on the target domain. While we develop the GFK and landmark based approaches, we have assumed that the source and the target domains are already given. However, this may be not the case in practice. It is the time for us to step back and investigate the *domains* now (in addition to the *adaptation*). *What exactly is a domain composed of? How are domains different from each other*? One pitfall is that the artificially collected datasets could be an agglomeration of several distinctive domains. Thus, modeling a dataset as a single domain would necessarily blend the distinctions, potentially damaging the discrimination in data and achieving only suboptimal performance at the test stage.

In this chapter, we propose an approach to automatically discover latent domains in the datasets. We show that reshaping data according to the discovered domains significantly improves the adaptation performance on the test sets. Our formulation imposes two key properties on domains: maximum distinctiveness and maximum learnability. By maximum distinctiveness, we require the underlying distributions of the identified domains to be different from each other to the maximum extent; by maximum learnability, we ensure that a strong discriminative model can be learned from the domain. We devise a nonparametric formulation and efficient optimization procedure that can successfully discover domains among both training and test data.

7.1 Motivation and main idea

A *domain* refers to an underlying data distribution. In most real applications, the distributions are unknown and instead we observe some data samples drawn from them. Then, which groups of data constitute which domains? For some applications, the answers come naturally. For example, in speech recognition, we can organize data into speaker-specific domains where each domain contains a single speaker's utterances. In language processing, we can organize text data into language-specific domains. For those types of data, we can neatly categorize each instance with a discrete set of *semantically meaningful* properties; a domain is thus naturally composed of instances of the same (subset of) properties.

For other applications, however, the same is not possible. *Take visual recognition for instance*. In addition to large intra-category appearance variations, images and video of objects (or scenes, attributes, activities, etc.) are also significantly affected by many extraneous factors such as pose, illumination, occlusion, camera resolution, and background. Many of these factors simply do not naturally lend themselves to deriving discrete domains. Furthermore, the factors overlap and

interact in images in complex ways. In fact, even coming up with a comprehensive set of such properties is a daunting task in its own right—not to mention automatically detecting them in images!

Partially due to these conceptual and practical constraints, datasets for visual recognition are not deliberately collected with clearly identifiable domains (Deng et al., 2009; Everingham et al., 2007; Russell et al., 2008; Griffin et al., 2007; Weinland et al., 2007). Instead, standard image/video collection is a product of trying to ensure coverage of the target category labels on one hand, and managing resource availability on the other. As a result, a troubling practice in visual domain adaptation research is to equate *datasets* with domains and study the problem of *cross-dataset generalization* or *correcting dataset bias* (Torralba & Efros, 2011; Gong et al., 2012a; Cao et al., 2010; Tommasi et al., 2012).

One pitfall of this ad hoc practice is that a dataset could be an agglomeration of several distinctive domains. Thus, modeling the dataset as a single domain would necessarily blend the distinctions, potentially damaging visual discrimination. Consider the following human action recognition task, which is also studied empirically in this work. Suppose we have a training set containing videos of multiple subjects taken at view angles of 30° and 90°, respectively. Unaware of the distinction of these two views of videos, a model for the training set as a single training domain needs to account for both inter-subject and inter-view variations. Presumably, applying the model to recognizing videos taken at view angle of 45° (*i.e.*, from the test domain) would be less effective than applying models accounting for the two view angles separately, i.e., modeling inter-subject variations only.

How can we avoid such pitfalls? More specifically, *how can we form characteristic domains*, without resorting to the hopeless task of manually defining properties along which to organize them? We propose novel learning methods to automatically reshape datasets into domains. This is a challenging unsupervised learning problem. At the surface, we are not given any information about the domains that the datasets contain, such as the statistical properties of the domains, or even the number of domains. Furthermore, the challenge cannot be construed as a traditional clustering problem; simply clustering images by their appearance is prone to reshaping datasets into per-category domains, as observed in (Hoffman et al., 2012) and our own empirical studies. Moreover, there may be many complex factors behind the domains, making it difficult to model the domains with parametric mixture models on which traditional clustering algorithms (e.g., Kmeans or Gaussian mixtures) are based.

Our key insights are two axiomatic properties that latent domains should possess: *maximum distinctiveness* and *maximum learnability*. By *maximum distinctiveness*, we identify domains that are maximally different in distribution from each other. This ensures domains are characteristic in terms of their large inter-domain variations. By *maximum learnability*, we identify domains from which we can derive strong discriminative models to apply to new testing data.

In Section 7.2, we describe our learning methods for extracting domains with these desirable properties. We derive nonparametric approaches to measuring domain discrepancies and show how to optimize them to arrive at maximum distinctiveness. We also show how to achieve maximum learnability by monitoring an extracted domain's discriminative learning performance. The learnability signals the number of latent domains underlying the datasets.

In Section 7.3, we show how to conditionally reshape the test data to different groups, each of which is matched to an identified domain as much as possible. We then run the domain adaptation algorithms upon the matched training-test pairs.

7.2 Discovering latent domains from the training data

We assume that we have access to one or more annotated datasets with a total of M data instances. The data instances are in the form of $(\boldsymbol{x}_m, y_m)$ where $\boldsymbol{x}_m \in \mathbb{R}^{\mathsf{D}}$ is the feature vector and $y_m \in [\mathsf{C}]$ the corresponding label out of C categories. Moreover, we assume that each data instance comes from a latent domain $z_m \in [\mathsf{K}]$ where K is the number of domains.

In what follows, we start by describing our algorithm for inferring z_m assuming K is known. Then we describe how to infer K from the data.

7.2.1 Maximally distinctive domains

Given K, we denote the distributions of unknown domains $\mathcal{D}_k$ by $P_k(\boldsymbol{x}, y)$ for $k \in [\mathsf{K}]$. We do not impose any parametric form on $P_k(\cdot, \cdot)$. Instead, the marginal distribution $P_k(\boldsymbol{x})$ is approximated by the empirical distribution $\hat{P}_k(\boldsymbol{x})$

$$\hat{P}_k(\boldsymbol{x}) = \frac{1}{\mathsf{M}_k} \sum_m \delta_{\boldsymbol{x}_m} z_{mk},$$

where M_k is the number of data instances to be assigned to the domain k and $\delta_{\boldsymbol{x}_m}$ is an atom at $\boldsymbol{x}_m$. $z_{mk} \in \{0, 1\}$ is a binary indicator variable and takes the value of 1 when $z_m = k$. Note that $\mathsf{M}_k = \sum_m z_{mk}$ and $\sum_k \mathsf{M}_k = \mathsf{M}$.

What kind of properties do we expect from $\hat{P}_k(\boldsymbol{x})$? Intuitively, we would like any two different domains $\hat{P}_k(\boldsymbol{x})$ and $\hat{P}_{k'}(\boldsymbol{x})$ to be as distinctive as possible. In the context of modeling visual data, this implies that intra-class variations between domains are often far more pronounced than inter-class variations within the same domain. As a concrete example, consider the task of differentiating commercial jetliners from fighter jets. While the two categories are easily distinguishable when viewed from the same pose (frontal view, side view, etc.), there is a significant change in appearance when either category undergoes a pose change. Clearly, defining domains by simply clustering the images by appearance is insufficient; the inter-category and inter-pose variations will both contribute to the clustering procedure and may lead to unreasonable clusters. Instead, to identify characteristic domains, we need to look for divisions of the data that yield *maximally distinctive* distributions.

To quantify this intuition, we need a way to measure the difference in distributions. To this end, we apply the kernel-based MMD method to examine whether two samples are from the same distribution (Gretton et al., 2006). Concretely, let $k(\cdot, \cdot)$ denote a characteristic positive semidefinite kernel (such as the Gaussian kernel) and $\phi(x) = k(\cdot, x)$. We compute the the difference between the means of two empirical distributions in the reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ induced by the kernel function,

$$d(k, k') = \left\| \frac{1}{\mathsf{M}_k} \sum_m \phi(\boldsymbol{x}_m) z_{mk} - \frac{1}{\mathsf{M}'_k} \sum_m \phi(\boldsymbol{x}_m) z_{mk'} \right\|^2_{\mathcal{H}} \tag{7.1}$$

where $\phi(\boldsymbol{x}_m)$ is the image (or kernel-induced feature) of $\boldsymbol{x}_m$ under the kernel. The measure approaches zero as the number of samples tends to infinity, if and only if the two domains are the

same, $P_k = P_{k'}$. We define the *total domain distinctiveness* (TDD) as the sum of this quantity over all possible pairs of domains:

$$\mathsf{TDD}(\mathsf{K}) = \sum_{k \neq k'} d(k, k'), \tag{7.2}$$

and choose domain assignments for z_m such that TDD is maximized.

Label prior constraint In addition to the binary constraints on z_{mk}, we also enforce

$$\sum_{k=1}^{\mathsf{K}} z_{mk} = 1, \quad \forall\, m \in [\mathsf{M}], \text{ and } \frac{1}{\mathsf{M}_k} \sum_{m=1}^{\mathsf{M}} z_{mk} y_{mc} = \frac{1}{\mathsf{M}} \sum_{m=1}^{\mathsf{M}} y_{mc}, \quad \forall\, c \in [\mathsf{C}], \quad k \in [\mathsf{K}] \tag{7.3}$$

where y_{mc} is a binary indicator variable, taking the value of 1 if $y_m = c$.

The first constraint stipulates that every instance will be assigned to one domain and one domain only. The second constraint, which we refer to as the *label prior constraint* (LPC), requires that within each domain, the class labels are distributed according to the prior distribution (of the labels), estimated empirically from the labeled data.

LPC does not restrict the absolute numbers of instances of different labels in each domain. It only reflects the intuition that in the process of data collection, the relative percentages of different classes are approximately in accordance with a prior distribution that is independent of domains. For example, in action recognition, if the "walking" category occurs relatively frequently in a domain corresponding to brightly lit video, we also expect it to be frequent in the darker videos. Thus, when data instances are re-arranged into latent domains, the same percentages are likely to be preserved.

The optimization problem (eq. (7.2) and eq. (7.3)) is NP-hard due to the integer constraints. In the following, we relax it into a continuous optimization, which is more accessible with off-the-shelf optimization packages.

Relaxation We introduce new variables $\beta_{mk} = z_{mk}/\mathsf{M}_k$, and relax them to live on the simplex

$$\boldsymbol{\beta}_k = (\beta_{1k}, \cdots, \beta_{\mathsf{M}k})^{\mathrm{T}} \in \Delta = \left\{ \boldsymbol{\beta}_k : \beta_{mk} \geq 0, \sum_{m=1}^{\mathsf{M}} \beta_{mk} = 1 \right\}$$

for $k = 1, \cdots, \mathsf{K}$. With the new variables, our optimization problem becomes

$$\max_{\boldsymbol{\beta}} \quad \sum_{k \neq k'} \mathsf{TDD}(\mathsf{K}) = \sum_{k \neq k'} (\boldsymbol{\beta}_k - \boldsymbol{\beta}_{k'})^{\mathrm{T}} \boldsymbol{K} (\boldsymbol{\beta}_k - \boldsymbol{\beta}_{k'}) \tag{7.4}$$

$$\text{s.t.} \quad 1/\mathsf{M} \leq \sum_k \beta_{mk} \leq 1/\mathsf{C}, \quad m = 1, 2, \cdots, \mathsf{M}, \tag{7.5}$$

$$(1-\delta)/\mathsf{M} \sum_m y_{mc} \leq \sum_m \beta_{mk} y_{mc} \leq (1+\delta)/\mathsf{M} \sum_m y_{mc}, \quad c = 1, \cdots, \mathsf{C}, \quad k = 1, \cdots, \mathsf{K},$$

where $\boldsymbol{K}$ is the $\mathsf{M} \times \mathsf{M}$ kernel matrix. The first constraint stems from the (default) requirement that every domain should have at least one instance per category, namely, $\mathsf{M}_k \geq \mathsf{C}$ and every domain should at most have M instances ($\mathsf{M}_k \leq \mathsf{M}$). The second constraint is a relaxed version of the LPC, allowing a small deviation from the prior distribution by setting $\delta = 1\%$. We assign $\boldsymbol{x}_m$ to the domain k for which β_{mk} is the maximum of $\beta_{m1}, \cdots, \beta_{m\mathsf{K}}$.

This relaxed optimization problem is a maximization of convex quadratic function subject to linear constraints. Though in general still NP-hard, this type of optimization problem has been studied extensively and we have found existing solvers are adequate in yielding satisfactory solutions.

7.2.2 Maximally learnable domains: determining the number of domains

Given M instances, how many domains hide inside? Note that the total domain distinctiveness $\mathsf{TDD}(\mathsf{K})$ increases as K increases — presumably, in the extreme case, each domain has only a few instances and their distributions would be maximally different from each other. However, such tiny domains would offer insufficient data to separate the categories of interest reliably.

To infer the optimal K, we appeal to *maximum learnability*, another desirable property we impose on the identified domains. Specifically, for any identified domain, we would like the data instances it contains to be adequate to build a strong classifier for labeled data — failing to do so would cripple the domain's adaptability to new test data.

Following this line of reasoning, we propose domain-wise cross-validation (DWCV) to identify the optimal K. DWCV consists of the following steps. First, starting from $\mathsf{K} = 2$, we use the method described in the previous section to identify K domains. Second, for each identified domain, we build discriminative classifiers, using the label information and evaluate them with cross-validation. Denote the cross-validation accuracy for the k-th domain by A_k. We then combine all the accuracies with a weighted sum

$$A(\mathsf{K}) = 1/\mathsf{M} \sum_{k=1}^{\mathsf{K}} \mathsf{M}_k A_k.$$

For very large K such that each domain contains only a few examples, $A(\mathsf{K})$ approaches the classification accuracy using the class prior probability to classify. Thus, starting at $\mathsf{K} = 2$ (and assuming $A(2)$ is greater than the prior probability's classification accuracy), we choose K^* as the value that attains the highest cross-validation accuracy: $\mathsf{K}^* = \arg\max_{\mathsf{K}} A(\mathsf{K})$. For N-fold cross-validation, a practical bound for the largest K we need to examine is $\mathsf{K}_{\max} \leq \min\{\mathsf{M}/(\mathsf{NC}), \mathsf{C}\}$. Beyond this bound it does not quite make sense to do cross-validation.

7.3 Conditionally reshaping the test data

Not only the training data but also the test datasets may be composed of several latent domains. However, the reshaping process of the test data has a critical difference from that of the training. Specifically, we should reshape the test data *conditioning* on the identified domains from the training datasets — the goal is to discover latent domains in the test datasets that match the domains in the training datasets as much as possible. We term this *conditional reshaping*.

Concretely, given the test data $\{(\boldsymbol{x}_n^t, y_n^t)\}_{n=1}^{\mathsf{N}}$ potentially drawn from multiple domains, we introduce indicator variables $z_{nv}^t \in \{0,1\}$, $n = 1, \cdots, \mathsf{N}$, $v = 1, \cdots, \mathsf{K}$ for the test data points, and solve for the variables by minimizing the empirical distance between the distribution embeddings of an identified training domain and an latent domain of the test datasets, respectively,

$$\min_{\{z_{nv}^t\}} \quad \sum_{v=1}^{\mathsf{K}} \left\| \frac{1}{\mathsf{M}_v} \sum_{m=1}^{\mathsf{M}} z_{mv} \phi(\boldsymbol{x}_m) - \frac{1}{\sum_{j=1}^{\mathsf{N}} z_{jv}^t} \sum_{n=1}^{\mathsf{N}} z_{nv}^t \phi(\boldsymbol{x}_n^t) \right\|_{\mathcal{H}}^2, \tag{7.6}$$

where the values of z_{mv}'s are inherited from Section 2 in the main text. Note that we do not impose any balance constraints on z_{nv}^t, allowing the number of test domains smaller than that of the training domains.

With $\beta_{nv}^t = z_{nv}^t / \sum_j z_{jv}^t$ we relax problem (7.6) to

$$\begin{aligned} \min_{\boldsymbol{\beta}^t} \quad & \sum_v {\boldsymbol{\beta}_v^t}^T K^t \boldsymbol{\beta}_v^t - \frac{2}{\mathsf{M}_v} \mathbf{1}_{\mathsf{M}_v}^T K^{vt} \boldsymbol{\beta}_v^t \\ \text{s.t.} \quad & \mathbf{0} \leq \boldsymbol{\beta}^t \leq \mathbf{1}, \quad \mathbf{1}^T \boldsymbol{\beta}_v^t = 1, \forall v, \quad \sum_v \beta_{nv}^t \leq 1, \forall n \end{aligned} \tag{7.7}$$

where $\boldsymbol{\beta}_v = (\beta_{1v}, \beta_{2v}, \cdots, \beta_{\mathsf{N}v})^T$, $K_{ij}^t = k(\boldsymbol{x}_i^t, \boldsymbol{x}_j^t)$, $K_{ij}^{vt} = k(\boldsymbol{x}_i^v, \boldsymbol{x}_j^t)$, and $\boldsymbol{x}_i^v$ is a training data point assigned to the v-th domain. Problem (7.7) is a convex quadratic programming problem which can be solved efficiently. We recover the indicator variables by $z_{nv^\star}^t = 1$ if $v^\star = \arg\max_v \beta_{nv}^t$.

After slicing the test datasets in this way, each slice/subset is also matched to a particular training domain in terms of the smallest distribution distance. Arguably, the adaptation between the matched pairs are easier than some arbitrary pairs or the original datasets.

7.4 Summary

We have introduced two axiomatic properties domains should possess, *maximum distinctiveness* and *maximum learnability*, to discover latent domains from datasets. Accordingly, we proposed nonparametric approaches encouraging the extracted domains to satisfy these properties. Since in each domain the data discrimination is more consistent than that in the heterogeneous datasets, stronger classifiers can be trained and better prediction performance can be achieved on the target domain. We have also shown how to conditionally reshape the test data to match different groups of them to the identified domains.

Part III

Kernels in Determinantal Point Process

Chapter 8

Sequential Determinantal Point Process and Video Summarization

Video summarization is a challenging problem with great application potentials. Whereas prior approaches, largely unsupervised in nature, focus on sampling useful frames and assembling them as summaries, we consider video summarization as a supervised subset selection problem. Our idea is to teach the system to learn from human-created summaries how to select informative and diverse subsets, so as to best meet evaluation metrics derived from human-perceived quality. To this end, we propose the *sequential determinantal point process* (seqDPP), a new probabilistic model for diverse sequential subset selection. Our novel seqDPP heeds the inherent sequential structures in video data, thus overcoming the deficiency of the standard DPP, which treats video frames as randomly permutable items. Meanwhile, seqDPP retains the power of modeling diverse subsets, essential for summarization. Our extensive results summarizing videos from 3 datasets demonstrate the superior performance of our method, compared to not only existing unsupervised methods but also naive applications of the standard DPP model.

8.1 Introduction

It is an impressive yet alarming fact that there is far more video being captured—by consumers, scientists, defense analysts, and others—than can ever be watched or browsed efficiently. For example, 144,000 hours of video are uploaded to YouTube daily; lifeloggers with wearable cameras amass Gigabytes of video daily; 422,000 CCTV cameras perched around London survey happenings in the city 24/7; UAV's patrol for 40 hours at a time collecting 27 Gigapixels per second. With this explosion of video data comes an ever-pressing need to develop *automatic video summarization* algorithms. By taking a long video as input and producing a short video (or keyframe sequence) as output, video summarization has great potential to reign in raw video and make it substantially more browseable and searchable.

Video summarization methods often pose the problem in terms of *subset selection*: among all the frames (subshots) in the video, which key frames (subshots) should be kept in the output summary? There is a rich literature in computer vision and multi-media developing a variety of ways to answer this question (Hong et al., 2009; Ngo et al., 2003; Ma et al., 2002; Liu & Kender, 2002; Pritch et al., 2007; Kang et al., 2006; Feng et al., 2012; Lee et al., 2012; Lu & Grauman, 2013; Khosla et al., 2013). Existing techniques explore a plethora of properties that a good summary should capture, designing criteria that the algorithm should prioritize when deciding which subset of frames (or subshots) to select. These include *representativeness* (the frames should depict the main contents of the videos) (Hong et al., 2009; Ngo et al., 2003; Khosla

et al., 2013), *diversity* (they should not be redundant) (Liu & Kender, 2002; Zhang et al., 1997), *interestingness* (they should have salient motion/appearance (Ngo et al., 2003; Ma et al., 2002; Kang et al., 2006) or trackable objects (Pritch et al., 2007; Liu et al., 2010; Feng et al., 2012)), or *importance* (they should contain important objects that drive the visual narrative) (Lee et al., 2012; Lu & Grauman, 2013).

Despite valuable progress in developing the desirable properties of a summary, prior approaches are impeded by their unsupervised nature. Typically the selection algorithm favors extracting content that satisfies criteria like the above (diversity, importance, etc.), and performs some sort of frame clustering to discover events. Unfortunately, this often requires some hand-crafting to combine the criteria effectively. After all, the success of a summary ultimately depends on human perception. Furthermore, due to the large number of possible subsets that could be selected, it is difficult to directly optimize the criteria *jointly* on the selected frames as a subset; instead, sampling methods that identify independently useful frames (or subshots) are common.

To address these limitations, we propose to consider video summarization as a *supervised* subset selection problem. The main idea is to use examples of human-created summaries—together with their original source videos—to teach the system how to select informative subsets. In doing so, we can escape the hand-crafting often necessary for summarization, and instead directly optimize the (learned) factors that best meet evaluation metrics derived from human-perceived quality. Furthermore, rather than independently select "high scoring" frames, we aim to capture the interlocked dependencies between a given frame and all others that could be chosen.

To this end, we propose the *sequential determinantal point process* (seqDPP), a new probabilistic model for sequential and diverse subset selection. The determinantal point process (DPP) has recently emerged as a powerful method for selecting a diverse subset from a "ground set" of items (Kulesza & Taskar, 2012), with applications including document summarization (Kulesza & Taskar, 2011b) and information retrieval tasks (Gillenwater et al., 2012a). However, existing DPP techniques have a fatal modeling flaw if applied to video (or documents) for summarization: they fail to capture their inherent sequential nature. That is, a standard DPP for summarization treats the inputs as *bags of randomly permutable items* agnostic to any temporal structure. Our novel seqDPP overcomes this deficiency, making it possible to faithfully represent the temporal dependencies in video data. At the same time, it lets us pose summarization as a supervised learning problem.

While learning how to summarize from examples sounds appealing, why should it be possible—particularly if the input videos are expected to vary substantially in their subject matter?[1] Unlike more familiar supervised visual recognition tasks, where test data can be reasonably expected to look like the training instances, a supervised approach to video summarization must be able to learn generic properties that transcend the specific content of the training set. For example, the learner can recover a "meta-cue" for representativeness, if the input features record profiles of the similarity between a frame and its increasingly distant neighbor frames. Similarly, category-independent cues about an object's placement in the frame, the camera person's active manipulation of viewpoint/zoom, etc., could play a role. In any such case, we can expect the learning algorithm to zero in on those meta-cues that are shared by the human-selected frames in the training set, even though the subject matter of the videos may differ.

[1] After all, not *all* videos on YouTube are about cats.

In short, our main contributions are: a novel learning model (seqDPP) for selecting diverse subsets from a sequence, its application to video summarization (the model is applicable to other sequential data as well), an extensive empirical study with three benchmark datasets, and a successful first-step/proof-of-concept towards using human-created summaries for learning to select subsets.

The rest of this chapter is organized as follows. In section 8.2, we review the basic concepts of DPP, its application to document summarization, and a large-margin parameter learning method. In section 8.3, we describe our seqDPP method, followed by the experimental results in section 8.5. We conclude this chapter by section 8.6.

8.2 Determinantal point process (DPP)

The DPP was first used in quantum physics to characterize the Pauli exclusion principle, which states that two identical particles cannot occupy the same quantum state simultaneously (Macchi, 1975). The ability to model exclusion/repulsion has made DPP an appealing tool in many other applications where diversity is preferred, including document summarization (Kulesza & Taskar, 2011b), or image search and ranking (Kulesza & Taskar, 2011c). For example, in the task of (extractive) document summarization, the goal is to generate a summary by selecting several sentences from a long document (Dang, 2005; Lin & Bilmes, 2010). A good summary needs to be both concise and informative of the original contents. As such, the selected sentences should be not only diverse (i.e., different) from each other to reduce the redundancy in the summary, but also representative of the topics in the document. To this end, Kulesza and Taskar applied DPP to this task by modeling abstractly how diverse items can be selected from a ground set. They achieved state-of-the-art performance on benchmark datasets (Kulesza & Taskar, 2011b, 2012).

In what follows, we give a brief account on DPP, how to apply it to document summarization, and how to learn DPP models by the large-margin principle we derived in (Chao et al., 2015).

8.2.1 Background

Given a ground set of M items, $\mathcal{Y} = \{1, 2, \ldots, \mathsf{M}\}$, a DPP defines a probabilistic measure over the power set, i.e., all possible subsets (including the empty set) of $\mathcal{Y}$. Concretely, let $\boldsymbol{L}$ denote a symmetric and positive semidefinite matrix in $\mathbb{R}^{\mathsf{M}\times\mathsf{M}}$. The probability of selecting a subset $\boldsymbol{y} \subseteq \mathcal{Y}$ is given by

$$P(\boldsymbol{y}; \boldsymbol{L}) = \det(\boldsymbol{L} + \boldsymbol{I})^{-1} \det(\boldsymbol{L}_{\boldsymbol{y}}), \tag{8.1}$$

where $\boldsymbol{L}_{\boldsymbol{y}}$ denotes the submatrix of $\boldsymbol{L}$, with rows and columns selected by the indices in $\boldsymbol{y}$. $\boldsymbol{I}$ is the identity matrix with the proper size. We define $\det(\boldsymbol{L}_{\emptyset}) = 1$. The above way of defining a DPP is called an L-ensemble. An equivalent way of defining a DPP is to use a kernel matrix to define the marginal probability of selecting a random subset:

$$P_{\boldsymbol{y}} = \sum_{\boldsymbol{y}' \subseteq \mathcal{Y}} P(\boldsymbol{y}'; \boldsymbol{L}) \mathbb{I}[\, \boldsymbol{y} \subseteq \boldsymbol{y}' \,] = \det(\boldsymbol{K}_{\boldsymbol{y}}), \tag{8.2}$$

where we sum over all subsets $\boldsymbol{y}'$ that contain $\boldsymbol{y}$ ($\mathbb{I}[\cdot]$ is an indicator function). The matrix $\boldsymbol{K}$ is another positive semidefinite matrix, computable from the $\boldsymbol{L}$ matrix

$$\boldsymbol{K} = \boldsymbol{L}(\boldsymbol{L} + \boldsymbol{I})^{-1}, \tag{8.3}$$

and $\boldsymbol{K}_{\boldsymbol{y}}$ is the submatrix of $\boldsymbol{K}$ indexed by $\boldsymbol{y}$. Despite the exponential number of summands in eq. (8.2), the marginalization is analytically tractable and computable in polynomial time.

Modeling Diversity One particularly useful property of the DPP is its ability to model *pairwise repulsion*. Consider the marginal probability of having two items i and j simultaneously in a subset:

$$\begin{aligned} P_{\{i,j\}} &= \det \begin{vmatrix} K_{ii} K_{ij} \\ K_{ji} K_{jj} \end{vmatrix} = K_{ii} K_{jj} - K_{ij}^2 \\ &\leq K_{ii} K_{jj} = P_{\{i\}} P_{\{j\}} \leq \min(P_{\{i\}}, P_{\{j\}}). \end{aligned} \tag{8.4}$$

Thus, unless $K_{ij} = 0$, the probability of observing i and j jointly is always less than observing either i or j separately. Namely, having i in a subset repulsively excludes j and vice versa. Another extreme case is when i and j are the same; then $K_{ii} = K_{jj} = K_{ij}$, which leads to $P_{\{i,j\}} = 0$. Namely, we should never allow them together in any subset.

Consequently, a subset with a large (marginal) probability cannot have too many items that are similar to each other (i.e., with high values of K_{ij}). In other words, the probability provides a gauge of the diversity of the subset. The most diverse subset, which balances all the pairwise repulsions, is the subset that attains the highest probability

$$\boldsymbol{y}^* = \arg\max_{\boldsymbol{y}} P(\boldsymbol{y}; \boldsymbol{L}). \tag{8.5}$$

Note that this MAP inference is computed with respect to the L-ensemble (instead of $\boldsymbol{K}$) as we are interested in the mode, not the marginal probability of having the subset. Unfortunately, the MAP inference is NP-hard (Ko et al., 1995). Various approximation algorithms have been investigated (Gillenwater et al., 2012b; Kulesza & Taskar, 2012).

8.2.2 Learning DPPs for document summarization

Suppose we model selecting a subset of sentences as a DPP over all sentences in a document. We are given a set of training samples in the form of documents (i.e., ground sets) and the ground-truth summaries. How can we discover the underlying parameter $\boldsymbol{L}$ so as to use it for generating summaries for new documents?

Note that the new documents will likely have sentences that have not been seen before in the training samples. Thus, the kernel matrix $\boldsymbol{L}$ needs to be reparameterized in order to generalize to unseen documents. (Kulesza & Taskar, 2011b) proposed a special reparameterization called quality/diversity decomposition:

$$\boldsymbol{L}_{ij} = q_i \phi_i^{\mathrm{T}} \phi_j q_j, \quad q_i = \exp\left(\frac{1}{2} \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{x}_i\right), \tag{8.6}$$

where ϕ_i is the normalized TF-IDF vector of the sentence i so that $\phi_i^{\mathrm{T}}\phi_j$ computes the cosine angle between two sentences. On the other hand, the "quality" feature vector $\boldsymbol{x}_i$ encodes the contextual information about i as well as its representativeness of other items. In document summarization, $\boldsymbol{x}_i$ are the sentence lengths, positions of the sentences in the texts, and other *meta cues*. The parameter $\boldsymbol{\theta}$ is then optimized with maximum likelihood estimation (MLE) such that the target subsets have the highest probabilities

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \sum_n \log P(Y = \boldsymbol{y}_n^*; \boldsymbol{L}_n(\boldsymbol{\theta})), \tag{8.7}$$

where $\boldsymbol{L}_n$ is the $\boldsymbol{L}$ matrix formulated using sentences in the n-th ground set, and $\boldsymbol{y}_n^*$ is the corresponding ground-truth summary.

8.2.3 Multiplicative Large-Margin DPPs

Maximum likelihood estimation does not closely track discriminative errors (Ng & Jordan, 2002; Vapnik, 2000; Jebara, 2012). While improving the likelihood of the ground-truth subset $\boldsymbol{y}_n^*$, MLE could also improve the likelihoods of other competing subsets. Consequentially, a model learned with MLE could have modes that very different subsets yet are very close to each other in their probability values. Having highly confusable modes is especially problematic for DPP's NP-hard MAP inference — the difference between such modes can fall within the approximation errors of approximate inference algorithms such that the true MAP cannot be easily extracted.

To address these deficiencies, we derive a large-margin based approach (Chao et al., 2015) aims to maintain or increase the margin between the correct subset and alternative, incorrect ones. Specifically, we formulate the following large margin constraints

$$\begin{aligned} \log P(\boldsymbol{y}_n^*; \boldsymbol{L}_n) &\geq \max_{\boldsymbol{y} \subseteq \mathcal{Y}_n} \log \ell(\boldsymbol{y}_n^*, \boldsymbol{y})\, P(\boldsymbol{y}; \boldsymbol{L}_n) \\ &= \max_{\boldsymbol{y} \subseteq \mathcal{Y}_n} \log \ell(\boldsymbol{y}_n^*, \boldsymbol{y}) + \log P(\boldsymbol{y}; \boldsymbol{L}_n), \end{aligned} \tag{8.8}$$

where $\ell(\boldsymbol{y}_n^*, \boldsymbol{y})$ is a loss function measuring the discrepancy between the correct subset and an alternative $\boldsymbol{y}$. We assume $\ell(\boldsymbol{y}, \boldsymbol{y}) = 0$.

Intuitively, the more different $\boldsymbol{y}$ is from $\boldsymbol{y}_n^*$, the larger the gap we want to maintain between the two probabilities. This way, the incorrect one has less chance to be identified as the most diverse one. Note that while similar intuitions have been explored in multiway classification and structured prediction, the margin here is *multiplicative* instead of additive — this is by design, as it leads to a tractable optimization over the exponential number of constraints, as we will explain later.

Design of the Loss Function A natural choice for the loss function is the Hamming distance between $\boldsymbol{y}_n^*$ and $\boldsymbol{y}$, counting the number of disagreements between two subsets:

$$\ell_H(\boldsymbol{y}_n^*, \boldsymbol{y}) = \sum_{i \in \boldsymbol{y}} \mathbb{I}[i \notin \boldsymbol{y}_n^*] + \sum_{i \notin \boldsymbol{y}} \mathbb{I}[i \in \boldsymbol{y}_n^*]. \tag{8.9}$$

In this loss function, failing to select the right item costs the same as adding an unnecessary item. In many tasks, however, this symmetry does not hold. For example, in summarizing a document, omitting a key sentence has more severe consequences than adding a (trivial) sentence.

To balance these two types of errors, we introduce the generalized Hamming loss function,

$$\ell_\omega(\boldsymbol{y}_n^*, \boldsymbol{y}) = \sum_{i \in \boldsymbol{y}} \mathbb{I}[i \notin \boldsymbol{y}_n^*] + \omega \sum_{i \notin \boldsymbol{y}} \mathbb{I}[i \in \boldsymbol{y}_n^*]. \tag{8.10}$$

When ω is greater than 1, the learning biases towards higher *recall* to select as many items in $\boldsymbol{y}_n$ as possible. When ω is significantly less than 1, the learning biases towards high *precision* to avoid incorrect items as much as possible. Our empirical studies demonstrate such flexibility and its advantages in two real-world summarization tasks (Chao et al., 2015).

Numerical optimization To overcome the challenge of dealing with an exponential number of constraints in eq. (8.8), we reformulate it as a tractable optimization problem. We first upper-bound the hard-max operation with Jensen's inequality (i.e., softmax):

$$\log P(\boldsymbol{y}_n^*; \boldsymbol{L}_n) \geq \log \sum_{\boldsymbol{y} \subseteq \mathcal{Y}} e^{\log \ell_\omega(\boldsymbol{y}_n^*, \boldsymbol{y}) \, P(\boldsymbol{y}; \boldsymbol{L}_n)} = \mathsf{softmax}_{\boldsymbol{y} \subseteq \mathcal{Y}_n} \log \ell_\omega(\boldsymbol{y}_n^*, \boldsymbol{y}) + \log P(\boldsymbol{y}; \boldsymbol{L}_n). \tag{8.11}$$

With the loss function $\ell_\omega(\boldsymbol{y}_n^*, \boldsymbol{y})$, the right-hand-side is computable in polynomial time,

$$\mathsf{softmax}_{\boldsymbol{y} \subseteq \mathcal{Y}_n} \log \ell_\omega(\boldsymbol{y}_n^*, \boldsymbol{y}) + \log P(\boldsymbol{y}; \boldsymbol{L}_n) = \log \left(\sum_{i \notin \boldsymbol{y}_n^*} K_{n_{ii}} + \omega \sum_{i \in \boldsymbol{y}_n^*} (1 - K_{n_{ii}}) \right), \tag{8.12}$$

where $K_{n_{ii}}$ is the i-th element on the diagonal of $\boldsymbol{K}_n$, the marginal kernel matrix corresponding to $\boldsymbol{L}_n$. The detailed derivation of this result is in the Appendix C. Note that $\boldsymbol{K}_n$ can be computed efficiently from $\boldsymbol{L}_n$ through the identity eq. (8.3).

The softmax can be seen as a summary of all undesirable subsets (the correct subset $\boldsymbol{y}_n$ does not contribute to the weighted sum as $\ell_\omega(\boldsymbol{y}_n^*, \boldsymbol{y}_n^*) = 0$). Our optimization balances this term with the likelihood of the target with the hinge loss function $[z]_+ = \max(0, z)$,

$$\min \sum_n \left[-\log P(\boldsymbol{y}_n^*; \boldsymbol{L}_n) + \lambda \log \left(\sum_{i \notin \boldsymbol{y}_n^*} K_{n_{ii}} + \omega \sum_{i \in \boldsymbol{y}_n^*} (1 - K_{n_{ii}}) \right) \right]_+, \tag{8.13}$$

where $\lambda \geq 0$ is a tradeoff coefficient, to be tuned on validation datasets. Note that this objective function subsumes maximum likelihood estimation where $\lambda = 0$. We optimize the objective function with subgradient descent. Details are in the Appendix D.

8.3 Sequential DPPs for supervised videos summarization

Despite its success in document summarization (Kulesza & Taskar, 2011b), a direct application of DPP to video summarization is problematic. The DPP model is agnostic about the order of the items. For video data (and to a large degree, text data), the model does not respect the inherent sequential structures. The second limitation is that the quality-diversity decomposition, while

cleverly leading to a convex optimization, limits the power of modeling complex dependencies among items. Specifically, only the quality factor q_i is optimized on the training data. We develop new approaches to overcome those limitations.

In what follows, we describe our approach for video summarization. Our approach contains three components: (1) a preparatory yet crucial step that generates ground-truth summaries from multiple human-created ones (section 8.3.1); (2) a new probabilistic model—the sequential determinantal point process (seqDPP)—that models the process of sequentially selecting diverse subsets (section 8.3.2); (3) a novel way of re-parameterizing seqDPP that enables learning more flexible and powerful representations for subset selection from standard visual and contextual features (section 8.3.3).

8.3.1 Generating ground-truth summaries

The first challenge we need to address is what to provide to our learning algorithm as ground-truth summaries. In many video datasets, each video is annotated (manually summarized) by multiple human users. While they were often well instructed on the annotation task, discrepancies are expected due to many uncontrollable individual factors such as whether the person was attentive, idiosyncratic viewing preferences, etc. There are some studies on how to evaluate automatically generated summaries in the presence of multiple human-created annotations (Valdés & Martínez, 2012; Dumont & Mérialdo, 2009; Li & Merialdo, 2010). However, for learning, our goal is to generate one single ground-truth or "oracle" summary per video.

Our main idea is to synthesize the oracle summary that maximally agrees with all annotators. Our hypothesis is that despite the discrepancies, those summaries nonetheless share the common traits of reflecting the subject matter in the video. These commonalities, to be discovered by our synthesis algorithm, will provide *strong enough* signals for our learning algorithm to be successful.

To begin with, we first describe a few metrics in quantifying the agreement in the simplest setting where there are only two summaries. These metrics will also be used later in our empirical studies to evaluate various summarization methods. Using those metrics, we then analyze the consistency of human-created summaries in two video datasets to validate our hypothesis. Finally, we present our algorithm for synthesizing one single oracle summary per video,

Evaluation metrics Given two video summaries A and B, we measure how much they are in agreement by first matching their frames, as they might be of different lengths. Following (Sandra et al., 2011), we compute the pairwise distances between all frames across the two summaries. Two frames are then "matched" if their visual difference is below some threshold; a frame is constrained to appear in the matched pairs at most once. After the matching, we compute the following metrics (commonly known as Precision, Recall and F-score):

$$P_{AB} = \frac{\#\text{matched frames}}{\#\text{frames in A}}, \quad R_{AB} = \frac{\#\text{matched frames}}{\#\text{frames in B}}, \quad F_{AB} = \frac{P_{AB} \cdot R_{AB}}{0.5(P_{AB} + R_{AB})}.$$

All of them lie between 0 and 1, and higher values indicate better agreement between A and B. Note that these metrics are not symmetric – if we swap A and B, the results will be different.

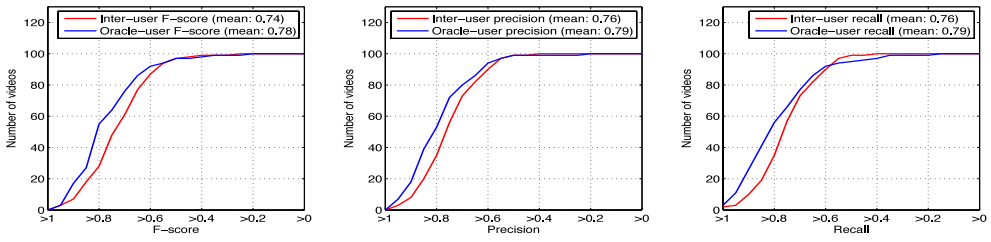

Figure 8.1: The agreement among human-created summaries is high, as is the agreement of the oracle summary generated by our algorithm (cf. section 8.3.1) and human annotations.

Our idea of examining the consistency among all summaries is to treat each summary in turn as if it were the gold-standard (and assign it as B) while treating the other summaries as A's. We report our analysis of existing video datasets next.

Consistency in existing video databases We analyze video summaries in two video datasets: 50 videos from the Open Video Project (OVP) (ope) and another 50 videos from Youtube (Sandra et al., 2011). Details about these two video datasets are in section 8.5. We briefly point out that the two datasets have very different subject matter and composition styles. Each of the 100 videos has 5 annotated summaries. For each video, we compute the pairwise evaluation metrics in precision, recall, and F-score by forming in total 20 pairs of summaries from two different annotators. We then average them per video. We plot how these averaged metrics distribute in Fig. 8.1. The plots show the number of videos (out of 100) whose averaged metrics exceed certain thresholds, marked on the horizontal axes. For example, more than 80% videos have an averaged F-score greater than 0.6, and 60% more than 0.7. Note that there are many videos ($\approx$20) with averaged F-scores greater than 0.8, indicating that on average, human-created summaries have a high degree of agreement. Note that the mean values of the averaged metrics per video are also high.

Greedy algorithm for synthesizing an oracle summary Encouraged by our findings, we develop a greedy algorithm for synthesizing one oracle summary per video, from multiple human-created ones. This algorithm is adapted from a similar one for document summarization (Kulesza & Taskar, 2011b).

For each video, we initialize the oracle summary with the empty set $\boldsymbol{y}^* = \emptyset$. Iteratively, we then add to $\boldsymbol{y}^*$ one frame i at a time from the video sequence

$$\boldsymbol{y}^* \leftarrow \boldsymbol{y}^* \cup \arg\max_i \sum_u F_{\boldsymbol{y}^* \cup i, \boldsymbol{y}_u}. \tag{8.14}$$

In words, the frame i is selected to maximally increase the F-score between the new oracle summary and the human-created summaries $\boldsymbol{y}_u$. To avoid adding all frames in the video sequence, we stop the greedy process as soon as there is no frame that can increase the F-score.

We measure the quality of the synthesized oracle summaries by computing their mean agreement with the human annotations. The results are shown in Fig. 8.1 too. The quality is high: more than 90% of the oracle summaries agree well with other summaries, with an F-score greater than

0.6. In what follows, we will treat the oracle summaries as ground-truth to inform our learning algorithms.

8.3.2 Sequential determinantal point processes (seqDPP)

The determinantal point process, as described in section 8.2, is a powerful tool for modeling diverse subset selection. However, video frames are more than items in a set. In particular, in DPP, the ground set is a bag – items are randomly permutable such that the most diverse subset remains unchanged. Translating this into video summarization, this modeling property essentially suggests that we could randomly shuffle video frames and expect to get the same summary!

To address this serious deficiency, we propose sequential DPP, a new probabilistic model to introduce strong dependency structures between items. As a motivating example, consider a video portraying the sequence of someone leaving home for school, coming back to home for lunch, leaving for market and coming back for dinner. If only visual appearance cues are available, a vanilla DPP model will likely select only one frame from the home scene and repel other frames occurring at the home. Our model, on the other hand, will recognize that the temporal span implies those frames are still *diverse* despite their visual similarity. Thus, our modeling intuition is that *diversity should be a weaker prior for temporally distant frames but ought to act more strongly for closely neighboring frames*. We now explain how our seqDPP method implements this intuition.

Model definition Given a ground set (a long video sequence) $\mathcal{Y}$, we partition it into T disjoint yet consecutive short segments $\bigcup_{t=1}^{T} \mathcal{Y}_t = \mathcal{Y}$. At time t, we introduce a subset section variable Y_t. We impose a DPP over two neighboring segments where the ground set is $\boldsymbol{U}_t = \mathcal{Y}_t \cup \boldsymbol{y}_{t-1}$, ie., the union between the video segments and the *selected subset* in the immediate past. Let $\boldsymbol{\Omega}_t$ denote the L-matrix defined over the ground set $\boldsymbol{U}_t$. The conditional distribution of Y_t is thus given by,

$$P(Y_t = \boldsymbol{y}_t | Y_{t-1} = \boldsymbol{y}_{t-1}) = \frac{\det \boldsymbol{\Omega}_{\boldsymbol{y}_{t-1} \cup \boldsymbol{y}_t}}{\det(\boldsymbol{\Omega}_t + \boldsymbol{I}_t)}. \tag{8.15}$$

As before, the subscript $\boldsymbol{y}_{t-1} \cup \boldsymbol{y}_t$ selects the corresponding rows and columns from $\boldsymbol{\Omega}_t$. $\boldsymbol{I}_t$ is a diagonal matrix, the same size as $\boldsymbol{U}_t$. However, the elements corresponding to $\boldsymbol{y}_{t-1}$ are zeros and the elements corresponding to $\mathcal{Y}_t$ are 1 (see (Kulesza & Taskar, 2012) for details). Readers who are familiar with DPP might identify the conditional distribution is also a DPP, restricted to the ground set $\mathcal{Y}_t$.

The conditional probability is defined in such a way that at time t, the subset selected should be diverse among $\mathcal{Y}_t$ as well as be diverse from previously selected $\boldsymbol{y}_{t-1}$. However, beyond those two priors, the subset is not constrained by subsets selected in the distant past. Fig. 8.2 illustrates the idea in graphical model notation. In particular, the joint distribution of all subsets is factorized

$$P(Y_1 = \boldsymbol{y}_1, Y_2 = \boldsymbol{y}_2, \cdots, Y_T = \boldsymbol{y}_T) = P(Y_1 = \boldsymbol{y}_1) \prod_{t=2} P(Y_t = \boldsymbol{y}_t | Y_{t-1} = \boldsymbol{y}_{t-1}). \tag{8.16}$$

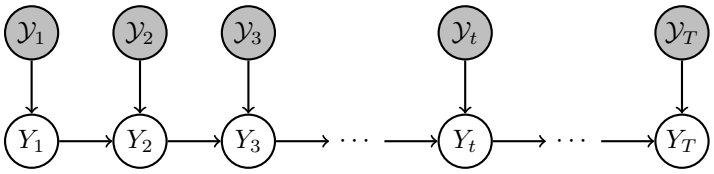

Figure 8.2: Our sequential DPP for modeling sequential video data, drawn as a Bayesian network

Inference and learning The MAP inference for the seqDPP model eq. (8.16) is as hard as the standard DPP model. Thus, we propose to use the following online inference, analogous to Bayesian belief updates (for Kalman filtering):

$$\boldsymbol{y}_1^* = \arg\max_{\boldsymbol{y}\in\mathcal{Y}_1} P(Y_1 = \boldsymbol{y}) \qquad \boldsymbol{y}_2^* = \arg\max_{\boldsymbol{y}\in\mathcal{Y}_2} P(Y_2 = \boldsymbol{y}|Y_1 = \boldsymbol{y}_1^*) \quad \cdots$$
$$\boldsymbol{y}_t^* = \arg\max_{\boldsymbol{y}\in\mathcal{Y}_t} P(Y_t = \boldsymbol{y}|Y_{t-1} = \boldsymbol{y}_{t-1}^*) \quad \cdots\cdots$$

Note that, at each step, the ground set could be quite small; thus an exhaustive search for the most diverse subset is plausible. The parameter learning is similar to the standard DPP model. We describe the details in the supplementary material.

8.3.3 Learning representations for diverse subset selection

As described previously, the kernel $\boldsymbol{L}$ of DPP hinges on the reparameterization with features of the items that can generalize across different ground sets. The quality-diversity decomposition in eq. (8.6), while elegantly leading to convex optimization, is severely limited in its power in modeling complex items and dependencies among them. In particular, learning the subset selection rests solely on learning the quality factor, as the diversity component remains handcrafted and fixed.

We overcome this deficiency with more flexible and powerful representations. Concretely, let $\boldsymbol{f}_i$ stand for the feature representation for item (frame) i, including both low-level visual cues and meta-cues such as contextual information. We reparameterize the $\boldsymbol{L}$ matrix with $\boldsymbol{f}_i$ in two ways.

Linear embeddings The simplest way is to linearly transform the original features

$$L_{ij} = \boldsymbol{f}_i^T \boldsymbol{W}^{\mathrm{T}} \boldsymbol{W} \boldsymbol{f}_j, \tag{8.17}$$

where $\boldsymbol{W}$ is the transformation matrix.

Nonlinear hidden representation We use a one-hidden-layer neural network to infer a hidden representation for $\boldsymbol{f}_i$

$$L_{ij} = \boldsymbol{z}_i^T \boldsymbol{W}^{\mathrm{T}} \boldsymbol{W} \boldsymbol{z}_j \quad \text{where} \quad \boldsymbol{z}_i = \tanh(\boldsymbol{U}\boldsymbol{f}_i), \tag{8.18}$$

where $\tanh(\cdot)$ stands for the hyperbolic transfer function.

To learn the parameters $\boldsymbol{W}$ or $\boldsymbol{U}$ and $\boldsymbol{W}$, we use maximum likelihood estimation (cf. eq. (8.7)), with gradient-descent to optimize. Details are given in the supplementary material.

Table 8.1: Performance of various video summarization methods on OVP. Ours and its variants perform the best.

	Unsupervised methods				Supervised subset selection			
	DT	STIMO	VSUMM$_1$	VSUMM$_2$	DPP + Q/D	Ours (seqDPP+)		
						Q/D	LINEAR	N.NETS
F	57.6	63.4	70.3	68.2	70.8±0.3	68.5±0.3	75.5±0.4	**77.7**±0.4
P	67.7	60.3	70.6	73.1	71.5±0.4	66.9±0.4	**77.5**±0.5	75.0±0.5
R	53.2	72.2	75.8	69.1	74.5±0.3	75.8±0.5	78.4±0.5	**87.2**±0.3

8.4 Related work

Space does not permit a thorough survey of video summarization methods. Broadly speaking, existing approaches develop a variety of selection criteria to prioritize frames for the output summary, often combined with temporal segmentation. Prior work often aims to retain diverse and representative frames (Ngo et al., 2003; Hong et al., 2009; Khosla et al., 2013; Liu & Kender, 2002; Zhang et al., 1997), and/or defines novel metrics for object and event saliency (Ma et al., 2002; Ngo et al., 2003; Kang et al., 2006; Lee et al., 2012). When the camera is known to be stationary, background subtraction and object tracking are valuable cues (e.g., (Pritch et al., 2007)). Recent developments tackle summarization for dynamic cameras that are worn or handheld (Khosla et al., 2013; Lee et al., 2012; Lu & Grauman, 2013) or design online algorithms to process streaming data (Feng et al., 2012).

Whereas existing methods are largely unsupervised, our idea to explicitly learn subset selection from human-given summaries is novel. Some prior work includes supervised learning *components* that are applied during selection (e.g., to generate learned region saliency metrics (Lee et al., 2012) or train classifiers for canonical viewpoints (Khosla et al., 2013)), but they do not train/learn the subset selection procedure itself. Our idea is also distinct from "interactive" methods, which assume a human is in the loop to give supervision/feedback on each individual *test* video (Ellouze et al., 2010; Goldman et al., 2006; Liu et al., 2010).

Our focus on the determinantal point process as the building block is largely inspired by its appealing property in modeling diversity in subset selection, as well as its success in search and ranking (Kulesza & Taskar, 2011c), document summarization (Kulesza & Taskar, 2011b), news headline displaying (Affandi et al., 2012), and pose estimation (Kulesza & Taskar, 2011a). Applying DPP to video summarization, however, is novel to the best of our knowledge.

Our seqDPP is closest in spirit to the recently proposed Markov DPP (Affandi et al., 2012). While both models enjoy the Markov property by defining conditional probabilities depending only on the immediate past, Markov DPP's ground set is still the whole video sequence, whereas seqDPP can select diverse sets from the present time. Thus, one potential drawback of applying Markov DPP is to select video frames out of temporal order, thus failing to model the sequential nature of the data faithfully.

8.5 Experiments

We validate our approach of sequential determinantal point processes (seqDPP) for video summarization on several datasets, and obtain superior performance to competing methods.

Table 8.2: Performance of our method with different representation learning

	VSUMM$_2$			seqDPP+LINEAR			seqDPP+N. NETS		
	F	P	R	F	P	R	F	P	R
Youtube	55.7	59.7	58.7	57.8±0.5	54.2±0.7	69.8±0.5	**60.3**±0.5	59.4±0.6	64.9±0.5
Kodak	68.91	75.7	80.6	75.3±0.7	77.8±1.0	80.4±0.9	**78.9**±0.5	81.9±0.8	81.1±0.9

8.5.1 Setup

Data We benchmark various methods on 3 video datasets: the Open Video Project (OVP), the Youtube dataset (Sandra et al., 2011), and the Kodak consumer video dataset (Luo et al., 2009). They have 50, 39[2], and 18 videos, respectively. The first two have 5 human-created summaries per video and the last dataset has one human-created summary per video. Thus, for the first two datasets, we follow the algorithm described in section 8.3.1 to create an oracle summary per video.

We follow the same procedure as in (Sandra et al., 2011) to preprocess the video frames. We uniformly sample one frame per second from a video and then apply two stages of pruning to remove uninformative frames. Details are in the supplementary material.

Features Each frame is encoded with a $\ell 2$-normalized 8192-dimensional Fisher vector $\boldsymbol{\phi}_i$ (Perronnin & Dance, 2007), computed from SIFT features (Lowe, 2004). The Fisher vector represents well the visual appearance of the video frame, and is hence used to compute the pairwise correlations of the frames in the quality-diversity decomposition (cf. eq. (8.6)). We derive the quality features $\boldsymbol{x}_i$ by measuring the representativeness of the frame. Specifically, we place a contextual window centered around the frame of interest, and then compute its mean correlation (using the SIFT Fisher vector) to the other frames in the window. By varying the size of the windows from 5 to 15, we obtain 12-dimensional contextual features. We also add features computed from the frame saliency map (Hou & Zhang, 2007).

To apply our method for learning representations (cf. section 8.3.3), however, we do not make a distinction between the two types, and instead compose a feature vector $\boldsymbol{f}_i$ by concatenating $\boldsymbol{x}_i$ and $\boldsymbol{\phi}_i$. The dimension of our linear transformed features $\boldsymbol{W}\boldsymbol{f}_i$ is 10, 40 and 100 for OVP, Youtube, and Kodak, respectively. For the neural network, we use 50 hidden units and 50 output units.

Other details For each dataset, we randomly choose 80% of the videos for training and use the remaining 20% for testing. We run 100 rounds of experiments and report the average performance, which is evaluated by the aforementioned F-score, Precision, and Recall (cf. section 8.3.1). For evaluation, we follow the standard procedure: for each video, we treat each human-created summary as golden-standard and assess the quality of the summary output by our algorithm. We then average over all human annotators to obtain the evaluation metrics for that video.

[2]In total there are 50 Youtube videos. We keep 39 of them after excluding the cartoon videos.

Figure 8.3: Exemplar video summaries results by our seqDPP LINEAR vs. VSUMM summary (Sandra et al., 2011).

8.5.2 Results

We contrast our approach to several state-of-the-art methods for video summarization—which include several leading unsupervised methods—as well as the vanilla DPP model that has been successfully used for document summarization but does not model sequential structures. We compare the methods in greater detail on the OVP dataset. Table 8.1 shows the results.

Unsupervised or supervised? The four unsupervised methods are DT (Mundur et al., 2006), STIMO (Furini et al., 2010), VSUMM_1 (Sandra et al., 2011), and VSUMM_2 with a postprocessing step to VSUMM_1 to improve the precision of the results. We implement VSUMM ourselves using features described in the orignal paper and tune its parameters to have the best test performance. All 4 methods use clustering-like procedures to identify key frames as video summaries. Results of DT and STIMO are taken from their original papers. They generally underperform VSUMM.

What is interesting is that the vanilla DPP does not outperform the unsupervised methods, despite its success in other tasks. On the other end, our supervised method seqDPP, when coupled with the linear or neural network representation learning, performs significantly better than all other methods.

We believe the improvement can be attributed to two factors working in concert: (1) modeling sequential structures of the video data (2) more flexible and powerful representation learning. This is evidenced by the rather poor performance of seqDPP with the quality/diversity (Q/D) decomposition, where the representation of the items is severely limited such that modeling temporal structures alone is simply insufficient.

Linear or nonlinear? Table 8.2 concentrates on comparing the effectiveness of these two types of representation learning. The performances of VSUMM are provided for reference only. We see

that learning representations with neural networks generally outperforms the linear representations.

Qualitative results We present exemplar video summaries results by different methods in Fig. 8.3. The challenging Youtube video illustrates the advantage of sequential diverse subset selection. The visual variance in the beginning of the video is far greater (due to close-shots of people) than that at the end (fading out). Thus the clustering-based VSUMM method is prone to select key frames from the first half of the video, collapsing the latter part. In contrast, our seqDPP copes with time-varying diversity very well. The Kodak video demonstrates again our method's ability in attaining high recall when users only make diverse selections locally but not globally. VSUMM fails to acknowledge temporally distant frames can be diverse despite their visual similarities.

8.6 Summary

Our novel learning model seqDPP is a successful first step towards using human-created summaries for learning to select subsets for challenging problems such as video summarization. We just scratched the surface of the fruit-bearing direction of supervised learning for subset selection. Many research questions are open and have potential to advance the state-of-the-art to the next level. For example, we plan to investigate how to learn even more powerful representations from low-level visual cues and other meta cues for video summarization.

Part IV

Experiments

Chapter 9

Experiments

We evaluate our methods in the context of text sentiment analysis, visual object recognition, and cross-view human action recognition. We compare them to baselines and other competing domain adaptation methods. The general experimental setup is presented in Section 9.1, including introduction to the learning tasks, benchmark datasets, and the feature representations of data.

We report first the recognition results of applying our geodesic flow kernel (GFK) approach (section 9.2), followed by the results of our landmark-based approach (section 9.3). While the landmark-based approach in general outperforms GFK on the benchmark datasets we have tested, we believe that the method of GFK can stand alone separate from the landmarks idea, making its results interesting and valuable in their own right. In particular, the kernel function can be used as a building block for other methods, as exemplified by our success with the landmark approach.

We also investigate other practical issues in applying domain adaptation techniques to real-world problems. In addition to improving the adaptation performance for a pair of *given* source and target domains, we also study how we can select which source domain to pair with the target domain using the proposed Rank-of-Domains (ROD), given multiple source domains and a target domain. We use ROD to rank a list of source domains based on how suitable they are to domain adaptation in Section 9.4.

As a novel application of the domain adaptation techniques, we investigate the *dataset bias* problem, recently studied in (Torralba & Efros, 2011). Through their analysis, the authors identified a few visual datasets of high "market value", suggesting that they are less biased, and more representative of real-world objects. We re-examine these datasets with a new perspective: *are such high-valued datasets indeed useful in improving a target domain's performance?* Our analysis suggests it would be beneficial to also consider "ease of adaptability" in assessing the value of datasets. We describe our findings in Section 9.5.

Finally, we report the results of identifying latent domains. Our experiments show that given a mono-domain test set, adapting from the identified domains significantly outperforms from the handcrafted domains. Note that the latter corresponds to several widely used benchmark datasets in domain adaptation. We also present the experimental results of reshaping heterogeneous test datasets into domains, by matching to the identified training domains.

9.1 Experimental setup

This section describes the general experimental setup, including learning tasks, benchmark datasets, and the feature representations of data.

Figure 9.1: Example images from the MONITOR category in Caltech-256, Amazon, DSLR, and Webcam. Caltech and Amazon images are mostly from online merchants, while DSLR and Webcam images are from offices. (Best viewed in color.)

9.1.1 Text sentiment analysis

Text sentiment analysis aims to detect or classify the attitude of a writer from the text. Such attitude may be the writer's affective state (*e.g.*, happy, angry, sad, *etc.*), intended communication goal that the writer wishes to convey to the readers, the judgement of the writer about something, and so on. It is of particular interest for merchants to automatically analyze the consumers' feedback about products or services.

In our experiments, we use a standard benchmark dataset on domain adaptation for the sentiment analysis task. It is a corpus of reviews for four types of products from Amazon: books, DVDs, electronics, and kitchen appliances (Blitzer et al., 2007). The reviews contain star ratings (1 to 5 stars). We follow the previous work and convert them to binary (*i.e.*, positive or negative) labels of the reviews. The learning task is to label each review as the positive or negative evaluation of the product.

Each type of products is taken as a domain and in each domain there are 1,000 positive and 1,000 negative reviews. We select the top 400 words of the largest mutual information with the labels as the dictionary to extract feature representations of the reviews. We then represent each review with a 400-dimensional vector of term counts (*i.e.*, bag-of-words). The vectors are normalized to have zero mean and unit standard deviation in each dimension.

9.1.2 Object recognition

We consider the object recognition task from images in this thesis. Given an image with an object, the system outputs a category label to predict which category this object belongs to.

Our experiments use the three datasets which were studied in (Saenko et al., 2010): Amazon (images downloaded from online merchants), Webcam (low-resolution images by a web camera), and DSLR (high-resolution images by a digital SLR camera). Additionally, to validate the proposed methods on a wide range of datasets, we add Caltech-256 (Griffin et al., 2007) as a fourth dataset. We regard each dataset as a domain.

We extracted 10 classes common to all four datasets: BACKPACK, TOURING-BIKE, CALCULATOR, HEAD-PHONES, COMPUTER-KEYBOARD, LAPTOP-101, COMPUTER-MONITOR, COMPUTER-MOUSE, COFFEE-MUG, and VIDEO-PROJECTOR. There are 8 to 151 samples per category per domain, and 2533 images in total. Fig. 9.1 highlights the differences among these domains with example images from the category of MONITOR.

We report our results on adapting 10-way classification[1] among those four domains. Our results of competing methods are based on rerunning publicly available codes of those methods or our implementation of them if the code is unavailable.

We follow similar feature extraction and experiment protocols used in previous work. Briefly, we use SURF features (Bay et al., 2006) and encode the images with 800-bin histograms with the codebook trained from a subset of Amazon images. The histograms are normalized first and then z-scored to have zero mean and unit standard deviation in each dimension. We are sharing our features (and code) publicly to promote direct reproducibility of our results[2].

For experiments using GFK for adaptation, we conduct experiments in 20 random trials for each pair of source and target domains. In each trial, we randomly sample labeled data in the source domain as training examples, and unlabeled data in the target domain as testing examples. This setting is in accordance with prior work (Saenko et al., 2010; Kulis et al., 2011; Gopalan et al., 2011) and provides the maximal comparability to those methods. More details on how data are split are provided in the next section. We report averaged accuracies on target domains as well as standard errors. For GFK results, 1-nearest neighbor is used as our classifier as it does not require cross-validating parameters. For our algorithms, the dimensionalities of subspaces are selected according to the criterion in section 4.4. For methods we compare to, we use what is recommended in the published work.

For experiments using our landmark-based approach for adaptation, we use all training instances from the source domains. Except this difference, other training setups are the same as for the experiments using GFK.

9.1.3 Cross-view human action recognition

For action recognition from video sequences, we use the IXMAS multi-view action dataset (Weinland et al., 2007). There are five views (Camera $0, 1, \cdots, 4$) of eleven actions in the dataset. Each action is performed three times by twelve actors and is captured by the five cameras. We keep the first five actions performed by *alba, andreas, daniel, hedlena, julien*, and *nicolas* such that the irregularly performed actions (Weinland et al., 2007) are excluded. In each view, 20 sequences are randomly selected per actor per action. We use the shape-flow descriptors to characterize the motion of the actions (Tran & Sorokin, 2008). Each view is taken as a domain.

9.2 Adaptation via the GFK

We conduct extensive evaluations of various baseline and competing approaches: (1) NO ADAPT where we use the original features, ie, without learning any new representations for domain adaptation; (2) PCA$_{\mathcal{S}}$ where we project the original features into the PCA subspace learned from the *source* domain; (3) PCA$_{\mathcal{T}}$ where we project the original features into the PCA subspace learned

[1] In the supplementary material for our previously published work (Gong et al., 2012b), we report our results on 31 categories common to Amazon, Webcam and DSLR, to compare directly to published results from the literature (Saenko et al., 2010; Kulis et al., 2011; Gopalan et al., 2011). Despite occasional discrepancies between the published results and the results obtained by our own experimentation on these 31 categories, they demonstrate the same trend—that our proposed methods significantly outperform competing approaches.

[2] `http://www-scf.usc.edu/~boqinggo/da.html`

Table 9.1: Recognition accuracies on target domains with *unsupervised* adaptation via GFK (C: Caltech, A: Amazon, W: Webcam, and D: DSLR).

Method	C→A	C→W	C→D	A→C	A→W	A→D	W→C	W→A	W→D	D→C	D→A	D→W
NO ADAPT	20.8±0.4	19.4±0.7	22.0±0.6	22.6±0.3	23.5±0.6	22.2±0.4	16.1±0.4	20.7±0.6	37.3±1.2	24.8±0.4	27.7±0.4	53.1±0.6
PCA$_{\mathcal{S}}$	34.7±0.5	*31.3*±0.6	33.6±1.2	34.0±0.3	31.3±0.5	29.4±0.8	23.4±0.6	28.0±0.5	68.2±1.0	26.8±0.3	28.1±0.3	61.7±0.7
PCA$_{\mathcal{T}}$	*37.5*±0.4	**33.9**±1.1	*37.8*±0.9	*35.4*±0.4	**34.9**±1.0	*33.3*±0.8	**29.6**±0.5	*32.5*±0.8	67.4±0.7	*31.2*±0.3	*34.4*±0.3	**79.4**±0.5
PCA$_{\mathcal{S}+\mathcal{T}}$	*36.6*±0.5	*32.1*±1.2	34.9±1.4	*35.8*±0.4	*32.8*±0.7	31.5±0.9	*28.1*±0.5	*31.6*±0.7	**74.1**±0.8	*30.8*±0.2	33.3±0.3	**79.7**±0.6
PLS$_{\mathcal{S}}$	26.7±0.9	26.0±0.6	28.2±1.3	31.1±0.5	29.3±0.9	28.0±1.0	18.3±0.5	21.1±0.9	42.8±1.4	21.4±0.6	26.5±0.6	41.9±1.4
GFS (impl.)	*36.8*±0.5	*30.6*±0.8	32.6±0.8	*35.3*±0.5	31.0±0.7	30.7±0.8	21.7±0.4	27.5±0.5	54.3±1.2	29.4±0.5	32.0±0.4	66.0±0.5
GFS (opti.)	*36.9*±0.5	**33.9**±1.2	35.2±1.0	*35.6*±0.4	**34.4**±0.9	**34.9**±0.9	*27.3*±0.5	*31.3*±0.7	*70.7*±0.9	30.0±0.2	32.6±0.5	*74.9*±0.6
GFK (A,A)	*36.9*±0.4	**33.7**±1.1	35.2±1.0	*35.6*±0.4	**34.4**±0.9	**35.2**±0.9	*27.2*±0.5	*31.1*±0.8	*70.6*±0.9	29.8±0.3	32.5±0.5	*74.9*±0.6
GFK (S,A)	**40.4**±0.7	**35.8±1.0**	**41.1**±1.3	**37.9**±0.4	**35.7**±0.9	**35.1**±0.8	**29.3**±0.4	**35.5**±0.7	*71.2*±0.9	**32.7**±0.4	**36.2**±0.4	**79.1**±0.7

from the *target* domain; (4) $\text{PCA}_{\mathcal{S}+\mathcal{T}}$ where we project the original features into the PCA subspace learned from the *combined* data from both the source and the target domains; (5) $\text{PLS}_{\mathcal{S}}$ where we project the original features into the Partial Least Squares (PLS) subspace computed using the source domain's labels. PLS is similar to PCA except it takes label information into consideration, and thus can be seen as a form of supervised dimensionality reduction (Hastie et al., 2009).

We also implement the method described in (Gopalan et al., 2011). We refer to it as the geodesic flow sampling approach (GFS). While it also uses geodesic flows to model domain mismatch between two domains, the approach *samples* a finite number of subspaces and uses them to construct high-dimensional features, followed by dimensionality reduction and classification. As the authors of this method suggest, we use PCA subspaces for both domains. We report results on two variants: i) our implementation using the recommended parameters reported in (Gopalan et al., 2011), such as the number of sampled subspaces and the reduced dimensionality (denoted GFS (impl.)), and ii) our implementation using the optimal dimensionality automatically selected by our algorithm (denoted GFS (opti.)).

For our approach, we use two types of subspaces for the source data: $\text{PCA}_{\mathcal{S}}$and $\text{PLS}_{\mathcal{S}}$. For the target domains, we use only $\text{PCA}_{\mathcal{T}}$ as there are no labels. Thus, there are two variants of our kernel-based method: GFK($\text{PCA}_{\mathcal{S}}$, $\text{PCA}_{\mathcal{T}}$) and GFK($\text{PLS}_{\mathcal{S}}$, $\text{PCA}_{\mathcal{T}}$).

9.2.1 Comparison results

Table 9.1 summarizes the classification accuracies as well as standard errors of all the above methods for different pairings of the source and target domains. Note that, to fit the table within the width of the page, we have shortened GFK($\text{PCA}_{\mathcal{S}}$, $\text{PCA}_{\mathcal{T}}$) to GFK(A,A), and GFK($\text{PLS}_{\mathcal{S}}$, $\text{PCA}_{\mathcal{T}}$) to GFK(S,A). The best group (differences up to one standard error) in each column are in bold font and the second best group (differences up to one standard error) are in italics and underlined.

All domain adaptation methods improve accuracy over the baseline NO ADAPT. Further, our GFK based methods in general outperform GFS. Moreover, GFK($\text{PLS}_{\mathcal{S}}$, $\text{PCA}_{\mathcal{T}}$) performs the best. Two key factors may contribute to the superiority of our method: i) the kernel integrates all the subspaces along the flow, and is hence able to model better the domain shift between the source and the target; ii) this method uses a discriminative subspace (by PLS) in the source domain to incorporate the label information. This has the benefit of avoiding projection directions that contain noise and very little useful discriminative information, albeit making source and target domains look similar. PCA, on the other hand, does not always yield subspaces that contain discriminative information. Consequently all the improvements by our GFK($\text{PLS}_{\mathcal{S}}$, $\text{PCA}_{\mathcal{T}}$) over GFS are statistically significant, with margins more than one standard error.

It is also interesting to note that the PCA-based baselines, especially $\text{PCA}_{\mathcal{S}+\mathcal{T}}$ and $\text{PCA}_{\mathcal{T}}$, perform quite well. They are often in the second-best performing group, and are even better than the GFS methods on DLSR $\rightarrow$ Webcam and Webcam $\rightarrow$ DSLR. We suspect that because the domain difference between DSLR and Webcam is small, either $\text{PCA}_{\mathcal{T}}$ or $\text{PCA}_{\mathcal{S}+\mathcal{T}}$ is already able to capture the commonness of the two domains well. For instance, both DSLR and Webcam contain similar office images though with different resolutions (see Fig. 9.1 for an example). The similarity between Webcam and DSLR is also confirmed by our ROD metric, which we will describe in section 9.4.

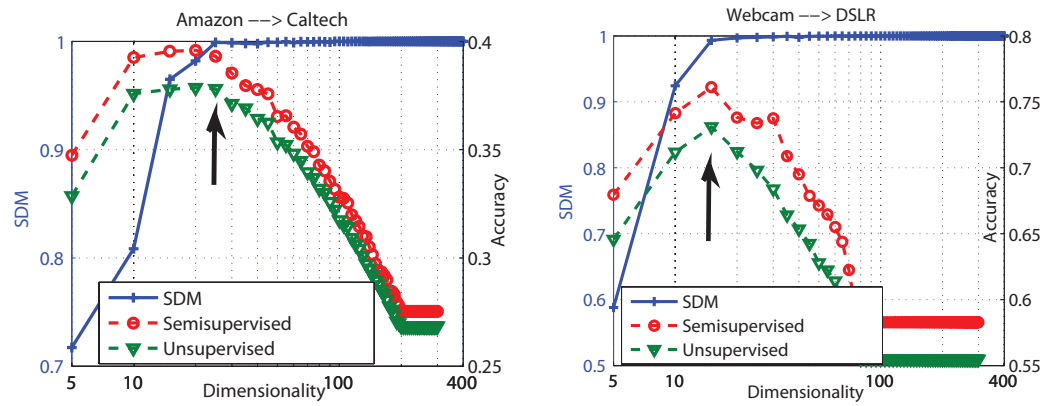

Figure 9.2: Selecting the optimal dimensionality d* with SDM (sec. 4.4); selected d* (where the arrows point to) leads to the best adaptation performance. (Best viewed in color)

9.2.2 Semi-supervised domain adaptation

In semi-supervised adaptation, we have access to a small labeled set of target data. It is straightforward to extend our GFK approach to take advantage of the labeled data. Concretely, we can construct GFK using the Partial Least Square subspace estimated on the target domain, i.e., GFK(PLS$_\mathcal{S}$, PLS$_\mathcal{T}$). We compare this approach to the metric learning based method METRIC (Saenko et al., 2010) which uses the correspondence between source and target labeled data to learn a Mahalanobis metric to map data into a new feature space.

Table 9.2 shows the results of all methods. Our GFK(PLS$_\mathcal{S}$, PCA$_\mathcal{T}$) is still the best, followed by GFK(PCA$_\mathcal{S}$, PCA$_\mathcal{T}$). Note that though GFK(PLS$_\mathcal{S}$, PLS$_\mathcal{T}$) incorporates discriminative information from both domains, it does not perform as well as GFK(PLS$_\mathcal{S}$, PCA$_\mathcal{T}$). This is probably due to the lack of enough labeled data in the target domains to give a reliable estimate of the PLS subspaces. The METRIC method does not perform well either, probably due to the same reason.

9.2.3 Automatic inferring the dimensionality of subspaces

Being able to choose the optimal dimensionality for the subspaces is an important property of our methods. Fig. 9.2 shows that the subspace disagreement measure (SDM) described in section 4.4 correlates well with recognition accuracies on the target domains. In the plots, the horizontal axis is the proposed dimensionality (in log scale) and the right vertical axis reports accuracies on both unsupervised domain adaptation and semi-supervised domain adaptation. The left vertical axis reports the values of SDM.

The plots reveal two conflicting forces at play. As the dimensionality increases, SDM—as a proxy to difference in geometric structures—quickly rises and eventually reaches its maximum value of 1. Beyond that point, adaptation becomes difficult as the subspaces have orthogonal directions.

However, before the maximum value is reached, the geometric difference is countered by the increase in variances — a small dimensionality would capture very little variances in the source domain data and would result in poor accuracies on both domains. The tradeoff occurs at where

Table 9.2: Recognition accuracies on target domains with *semi-supervised* adaptation via GFK (C: Caltech, A: Amazon, W: Webcam, and D: DSLR).

Method	C→D	C→W	C→A	A→C	A→W	A→D	W→C	W→A	W→D	D→C	D→A	D→W
NO ADAPT	26.5±0.7	25.2±0.8	23.1±0.4	24.0±0.3	31.6±0.6	28.1±0.6	20.8±0.5	30.8±0.6	44.3±1.0	22.4±0.5	31.3±0.7	55.5±0.7
PCA$_{\mathcal{S}}$	*48.9*±1.0	*54.2*±0.9	40.3±0.4	35.5±0.5	47.3±0.7	*47.8*±1.0	28.1±0.8	38.2±0.6	72.1±0.8	27.0±0.5	36.8±0.5	64.4±0.7
PCA$_{\mathcal{T}}$	*49.9*±0.8	52.1±0.8	*41.7*±0.4	*37.6*±0.4	51.8±0.8	44.1±1.0	**33.9**±0.6	41.5±0.5	70.0±0.7	**34.1**±0.4	*42.1*±0.4	*81.3*±0.4
PCA$_{\mathcal{S}+\mathcal{T}}$	*48.7*±1.2	*55.8*±0.9	*42.0*±0.6	*37.7*±0.4	49.8±1.0	*47.5*±1.2	**33.6**±0.7	*42.9*±0.6	**77.1**±0.6	**34.0**±0.4	*42.9*±0.5	**83.0**±0.4
PLS$_{\mathcal{S}}$	43.1±1.0	45.9±1.0	36.8±0.5	31.4±0.6	41.4±0.9	45.5±1.1	24.7±0.7	32.2±0.9	49.1±0.9	26.0±0.8	34.5±0.4	49.4±1.2
PLS$_{\mathcal{T}}$	27.3±1.1	25.3±0.4	28.9±0.6	26.3±0.3	23.6±0.9	28.0±1.0	22.2±0.4	25.2±0.9	47.0±1.2	25.8±0.4	27.9±0.4	47.1±0.9
PLS$_{\mathcal{S}+\mathcal{T}}$	36.9±0.9	37.0±0.9	33.5±0.5	32.4±0.4	35.6±1.1	36.9±1.2	25.4±0.8	31.6±0.6	52.1±1.2	27.5±0.7	32.9±0.6	53.1±1.2
METRIC (impl.)	35.0±1.1	34.7±1.0	33.7±0.8	27.3±0.7	36.0±1.0	33.7±0.9	21.7±0.5	32.3±0.8	51.3±0.9	22.5±0.6	30.3±0.8	55.6±0.7
GFS(impl.)	36.6±0.8	37.2±0.9	40.2±0.7	*37.7*±0.5	37.9±0.7	34.5±1.1	29.2±0.7	38.2±06	60.6±1.0	30.2±0.7	39.2±0.7	69.5±0.9
GFS(opti.)	*50.2*±0.8	*54.2*±0.9	*42.0*±0.5	*37.5*±0.4	*54.2*±0.8	*46.9*±1.1	**32.9**±0.7	*43.0*±0.7	*75.2*±0.7	*32.9*±0.4	**44.9**±0.7	78.6±0.4
GFK(A,A)	*49.5*±0.9	*54.2*±0.9	*42.0*±0.5	*37.8*±0.4	*53.7*±0.8	*47.0*±1.2	**32.8**±0.7	*42.8*±0.7	*75.0*±0.7	*32.7*±0.4	**45.0**±0.7	78.7±0.5
GFK(S,A)	**55.0**±0.9	**57.0**±0.9	**46.1**±0.6	**39.6**±0.4	**56.9**±1.0	**50.9**±0.9	*32.3*±0.6	**46.2**±0.7	*74.1*±0.9	**33.9**±0.6	**46.2**±0.6	80.2±0.4
GFK(S,S)	38.6±1.4	34.0±0.9	38.7±0.6	36.6±0.4	36.3±0.9	34.1±1.0	28.6±0.6	36.3±0.5	68.6±1.0	*32.6*±0.4	35.0±0.4	74.6±0.5

the geometric difference is just being maximized, justifying our dimensionality selection criterion in eq. (4.9).

9.3 Adaptation via the landmark approach

Next we test our landmark adaptation approach. There are several hyper-parameters in the landmark approach. The threshold of β_m in eq. (5.4) where we solve to select landmarks is set to be a small number (10^{-8}–10^{-10}) due to floating point arithmetics. The RBF kernel bandwidths in eq. (5.5) are $\sigma_q = 2^q\sigma_0$ with $q \in \{-6, -5, \cdots, 5, 6\}$, where σ_0 is the median of pairwise distances among all training instances. This ensures we select at least one instance per category and we do not select all instances in a category from the source domains as landmarks. The SVM tradeoff parameters are tuned on the validation data, cf. section 5.4. In general, our experimental results are robust to setting those parameters as long as we follow those mild guidelines.

9.3.1 Comparison results

Table 9.3 reports object recognition accuracies on the *target* under nine pairs of source and target domains[3]. We contrast the proposed approach (LANDMARK) to the methods of transfer component analysis (TCA) (Pan et al., 2009), geodesic flow sampling (GFS) (Gopalan et al., 2011), our GFK approaches for inferring invariant features and then classifying with 1-NN and linear SVM (GFK + 1NN and GFK + SUM), structural correspondence learning (SCL) (Blitzer et al., 2006), kernel mean matching (KMM) (Huang et al., 2006), and a metric learning method (METRIC) (Saenko et al., 2010) for *semi-supervised* domain adaptation, while label information (1 instance per category) from the target domains is used. We also report the baseline results of NO ADAPT, where we use source-only data and the original features to train classifiers.

Our approach LANDMARK clearly performs the best on almost all pairs, even when compared to the METRIC method which has access to labels from the target domains. The only significant exception is on the pair WEBCAM $\rightarrow$ DSLR. Error analysis reveals that the two domains are very similar, containing images of the same object instance with different imaging resolutions. As such, many data points in WEBCAM have been selected as landmarks, leaving very few instances for model selection during the discriminative training. Addressing this issue is left for future work.

Sentiment analysis Next, we report experimental results on the task of cross-domain sentiment analysis of text. We use the Amazon dataset described in (Blitzer et al., 2007). The dataset consists of product reviews on kitchen appliances, DVDs, books, and electronics. There are 1000 positive and 1000 negative reviews on each product type, each of which serves as a domain. We reduce the dimensionality to use the top 400 words which have the largest mutual information with the labels. We have found this preprocessing does not reduce performance significantly, while being computationally advantageous. We use bag-of-words as features.

In Table 9.4, we compare our LANDMARK method to leading methods for domain adaptation, including TCA (Pan et al., 2009), GFS (Gopalan et al., 2011), GFK (Gong et al., 2012b),

[3]We did not use DSLR as the source domain in these experiments as it is too small to select landmarks.

Table 9.3: Recognition accuracies on 9 pairs of unsupervised domain adaptation via the landmark approach. C: CALTECH, A: AMAZON, W: WEBCAM, D: DSLR. The proposed method (GFK+LANDMARK) performs the best on 8 out of 9 pairs, among all unsupervised methods.

%	A→C	A→D	A→W	C→A	C→D	C→W	W→A	W→C	W→D
NO ADAPT	41.7	41.4	34.2	51.8	54.1	46.8	31.1	31.5	70.7
TCA (Pan et al., 2009)	35.0	36.3	27.8	41.4	45.2	32.5	24.2	22.5	80.2
GFS (Gopalan et al., 2011)	39.2	36.3	33.6	43.6	40.8	36.3	33.5	30.9	75.7
GFK+1NN (OURS) (ours)	42.2	42.7	40.7	44.5	43.3	44.7	31.8	30.8	75.6
GFK+SVM (OURS)	38.8	43.3	37.3	50.2	40.1	45.1	39.1	34.5	67.5
SCL (Blitzer et al., 2006)	42.3	36.9	34.9	49.3	42.0	39.3	34.7	32.5	**83.4**
SCL (Blitzer et al., 2006)	42.3	36.9	34.9	49.3	42.0	39.3	34.7	32.5	83.4
KMM (Huang et al., 2006)	42.2	42.7	42.4	48.3	53.5	45.8	31.9	29.0	72.0
METRIC (Saenko et al., 2010)	42.4	42.9	**49.8**	46.6	47.6	42.8	38.6	33.0	**87.1**
GFK + LANDMARK (ours)	**45.5**	**47.1**	**46.1**	**56.7**	**57.3**	**49.5**	**40.2**	**35.4**	75.2

Table 9.4: Sentiment classification accuracies on target domains. K: KITCHEN, D: DVD, B: BOOKS, E: ELECTRONICS

%	K→D	D→B	B→E	E→K
NO ADAPTATION	72.7	73.4	73.0	81.4
TCA	60.4	61.4	61.3	68.7
GFS	67.9	68.6	66.9	75.1
GFK	69.0	71.3	68.4	78.2
SCL	72.8	76.2	75.0	82.9
KMM	72.2	78.6	76.9	**83.5**
METRIC	70.6	72.0	72.2	77.1
LANDMARK (**ours**)	**75.1**	**79.0**	**78.5**	**83.4**

SCL (Blitzer et al., 2006), KMM (Huang et al., 2006), METRIC (Saenko et al., 2010), as well as the baseline NO ADAPTATION.

Note that while SCL and KMM improve over the baseline, the other three methods underperform. Nonetheless, our method outperforms almost all other methods. Most interestingly, our method improves GFK significantly. We attribute its advantages to two factors: using multiple scales to analyze distribution similarity while GFK uses the "default" scale, and using landmarks to *discriminatively* learn invariant features.

It is also interesting to point out that, on the pair ELECTRONICS→KITCHEN, the improvements of all the domain adaptation methods over NO ADAPTATION are limited (at most 2%). This observation may direct some future work on studying under what situations adaptation techniques help, and how we can automatically determine such situations.

9.3.2 Detailed analysis on landmarks

Next, we further examine the utility of landmarks in domain adaptation to better understand why they are working as well as they do. We first study whether *automatically selected* landmarks coincide with our modeling intuition, ie, that they look like samples from the target domain.

Fig. 9.3 confirms our intuition. It displays several landmarks selected from the source domain AMAZON when the target domain is WEBCAM. The top-left panels display representative images from the HEADPHONE and MUG categories from WEBCAM, and the remaining panels display images from AMAZON, including both landmarks and those which are not.

When the scale σ is large, the selected landmarks are very similar in visual appearance to the representative images. As the scale decreases, landmarks with greater variance start to show. This is particularly pronounced at $2^{-3}\sigma_0$. Nonetheless, they still look far more likely to be from the target WEBCAM domain than non-landmark images (see bottom-right panels). Note that the non-landmark images for the HEADPHONE category contain images such as earphones, or headphones in packaging boxes. Similarly, non-landmark images in the MUG category are more unusually shaped ones.

Component-wise analysis In Table 9.5, we contrast our method to some of its variants, illustrating quantitatively the novelty and significance of using landmarks to facilitate adaptation.

Figure 9.3: Landmarks selected from the source domain AMAZON for the target domain WEBCAM, as well as non-landmarks (best viewed in color). As the scale decreases, images with greater variance in appearance are selected, as expected.

Table 9.5: Contrasting LANDMARK to several variants, illustrating the importance of our landmark selection algorithm.

%	A→C	A→D	A→W	C→A	C→D	C→W	W→A	W→C	W→D
GFK + LANDMARK (ours)	**45.5**	47.1	46.1	**56.7**	**57.3**	49.5	**40.2**	**35.4**	**75.2**
RAND. SEL.	44.5	44.5	41.9	53.8	49.9	49.5	39.8	34.1	74.2
SWAP	41.3	**47.8**	37.6	46.2	42.0	46.1	38.2	32.2	70.1
UNBALANCED	37.0	36.9	38.3	55.3	49.0	**50.1**	39.4	34.9	73.9
EUC + LANDMARK	44.5	44.0	41.0	50.2	40.1	45.1	39.1	34.5	67.5

First, we study the adverse effect of selecting incorrect images as landmarks. The row of RAND. SEL. displays results of randomly selecting landmarks, as opposed to using the algorithm proposed in Chapter 5. (The number of random landmarks is the average number of "true" landmarks chosen in LANDMARK.) The averaged accuracies over 10 rounds are reported. GFK+LANDMARK outperforms the random strategy, often by a significant margin, validating the automatic selection algorithm.

The SWAP row in Table 9.5 gives yet another strong indication of how landmarks could be viewed as samples from the target. Recall that landmarks are used as *training* data in the final stage of our learning algorithm to infer the domain-invariant feature space (cf. section 5.4). Other instances, ie, non-landmarks in the source, are used for model selection. This setup follows the intuition that as landmarks are mostly similar to the target, they are a better proxy than non-landmarks for optimizing discriminative loss for the target.

When we swap the setup, the accuracies drop significantly, except on the pair $A \rightarrow D$ (compare the rows SWAP and GFK+LANDMARK). This once again establishes the unique and extremely valuable role of landmarks.

We also study the usefulness of the class balancing constraint in eq. (5.3), which enforces that the selected landmarks obey the class prior distribution. Without it, some classes dominate and would result in poor classification results on the target domain. This is clearly evidenced in the row of UNBALANCED where accuracies drop significantly after we remove the constraint.

Finally, we study the effect of using GFK to measure distribution similarity, as in eq. (5.5). The row of EUC. + LANDMARK reports the results of using the conventional Euclidean distance, illustrating the striking benefit of using GFK (in the row of GFK+LANDMARK). While using nonparametric two-sample tests to measure distribution similarity has been previously used for domain adaptation (e.g., kernel mean matching, cf. the row of KMM in Table 9.3), selecting a proper kernel has received little attention, despite its vital importance. Our comparison to EUC. SEL. indicates that measuring distribution similarity *across* domains is greatly enhanced with a kernel revealing domain-invariant features.

Value of auxiliary tasks The auxiliary tasks are domain adaptation problems over new pairs of source and target domains $\mathcal{S}^q \rightarrow \mathcal{T}^q$, cf. section 5.3. As indicated by Theorem 1, by incorporating landmarks in the augmented target domain, the domain adaptation problem becomes easier to solve.

Fig. 9.4 provides strong empirical evidence supporting the theorem. In the figure, we show the object recognition accuracies on the original target domain as a result of solving those auxiliary tasks individually. Specifically, for each scale σ_q, we use the method of GFK to compute $\boldsymbol{G}_q$ for the pair $\mathcal{S}^q \rightarrow \mathcal{T}^q$ to extract invariant features then train a SVM classifier to minimize classification errors on the landmarks. We contrast to GFK+SVM reported in Table 9.3, where the only difference is to solve the original adaptation problem.

Clearly, the auxiliary tasks are easier to solve, resulting in more effective adaptations such that the accuracies on the target domains are in general much better than GFK+SVM. This asserts firmly that landmarks bridge between the source and the target, and thus are an important adaptation mechanism to exploit.

Benefits of multi-scale analysis and combining In Fig. 9.4, we also contrast results of individual tasks to the proposed method LANDMARK where the solutions of multiple auxiliary tasks

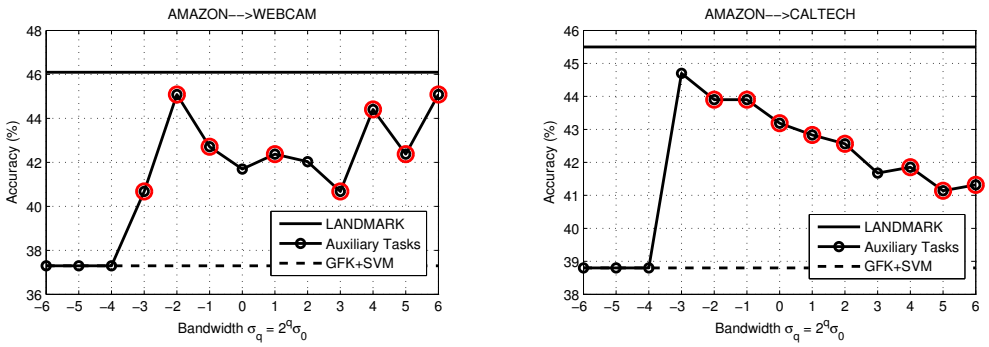

Figure 9.4: Performance of individual auxiliary tasks. The marked circle points on the curves show recognition accuracies on the original target domain $\mathcal{T}$, by using the kernel computed for the auxiliary task. Individual auxiliary tasks do not perform as well as LANDMARK. However, they all outperform GFK + SUM except when the scale is very small. In that case, all source domain data are selected as landmarks and auxiliary tasks are not defined. The red circles denote the auxiliary tasks whose kernels contribute to the final kernel $\boldsymbol{F}$ in eq. (5.7) after discriminative learning.

Table 9.6: ROD values between 4 domains. Lower values signify stronger adaptability of the corresponding source domain.

$\rightarrow$	CALTECH	AMAZON	DSLR	WEBCAM
CALTECH	0	**0.003**	0.21	0.09
AMAZON	**0.003**	0	0.26	0.05
DSLR	0.21	0.26	0	**0.03**
WEBCAM	0.09	0.05	**0.03**	0

are *combined* discriminatively. Combination clearly improves individual tasks. Moreover, we also mark in red color those individual tasks whose kernels have contributed to the final solution in eq. (5.7). Note that, the selected scales are indeed sparse. Both observations support our hypothesis that the data is modeled better with distances and similarities at multiple scales.

9.4 Which source domain should we use to adapt?

Now we examine whether the ROD metric (*cf.* Chapter 6) correlates with our empirical findings. We compute ROD using PCA subspaces and report the values among the four domains in Table 9.6. In general, ROD correlates well with recognition accuracies on the target domains and can reliably identify the best source domains to adapt. For example, when CALTECH is the target domain (the first column), AMAZON has the smallest value and AMAZON indeed leads to better classification accuracies on CALTECH than DSLR or WEBCAM. We find that the ROD metric also corroborates strongly with recognition accuracies for *semi-supervised* domain adaptation, cf. Table 9.2 in the Appendix. This further supports the value of the ROD metric as a barometer indicating whether two datasets are intrinsically similar, in both geometrical and statistical properties.

If we group CALTECH and AMAZON into a meta-category "Online" and DSLR and WEBCAM into another meta-category "Office", the distributions of ROD values with respect to the categories suggest that the domains with the same meta-category have stronger similarity than domain pairs crossing categories (such as CALTECH and DSLR). Thus ROD can also be used as a measure to partition datasets into clusters, where datasets in the same cluster share latent properties that might be of surprise to their users — the presence of such properties is probably not by design.

9.5 Ease of adaptation: a new perspective on datasets?

Torralba and Efors study the sources of dataset bias and the problem of cross-dataset generalization in several popular ones for object recognition (Torralba & Efros, 2011). To quantify the quality of each dataset, they devise a "market value" metric. Datasets with higher values are more diverse, and therefore are likely to reflect better the richness of real-world objects. In particular, they point out that PASCAL VOC 2007 and ImageNet have high values. However, we hypothesize that the market values could in some cases be overly pessimistic, since in their study no attempts were made to explicitly account for domain shifts between datasets.

Thus, building on their findings, we turn the tables around and investigate: *how valuable are these datasets in improving a target domain's performance?*

Table 9.7 summarizes our results on a subset of datasets used in (Torralba & Efros, 2011); PASCAL VOC 2007 (Everingham et al., 2007), ImageNet (Deng et al., 2009), and Caltech-101 (Fei-Fei et al., 2007). The recognition tasks are to recognize the category *person* and *car*. The cross-dataset generalization results are shown on the left side of the table, without using adaptation techniques (as in (Torralba & Efros, 2011)); and the adaptation results using our kernel-based method are on the right side of the table.

The rows are the source domain datasets and the columns are the target domains. The "Drop" columns report the percentages of drop in recognition accuracies between the source and the averaged accuracy on target domains, ie, the "Mean Targets" columns. The rightmost "Improvement" column is the percentage of improvement on target domains due to the use of domain adaptation. Clearly, domain adaptation noticeably improves recognition accuracies on the target domains. Caltech-101 is the exception where the improvement is marginal (47% vs. 46%). This corroborates the low "market value" assigned to this dataset in (Torralba & Efros, 2011).

PASCAL VOC 2007 has the smallest drop without domain adaptation so it would appear to be a better dataset than the other two. Once we have applied domain adaptation, we observe a negative drop — ie, the performance on the target domains is better than on the source domain itself! However, its improvement is not as high as ImageNet's.

Our conjecture is that the data in PASCAL VOC 2007 can be partitioned into two parts: one part is especially "hard" to be adapted to other domains and the other part is relatively "easy". The reverse of the performance drop suggests that the "easy" portion can be harvested by domain adaptation techniques. However, the benefit is limited due to the "hard" part. On the other end, for ImageNet, a larger portion of its data is perhaps amenable to adaptation. Hence, it attains a bigger improvement after adaptation.

In short, while PASCAL VOC 2007 and ImageNet are assigned the same "market value" in (Torralba & Efros, 2011), their usefulness to building object recognition systems that can be applied to other domains needs to be carefully examined in the context of adaptation. It might be

Table 9.7: Cross-dataset generalization with and without domain adaptation among domains with high and low "market values" (Torralba & Efros, 2011)

%	No domain adaptation					Using domain adaptation					
→	P	I	C101	Mean Targets	Drop_1	P	I	C101	Mean Targets	Drop_2	Improvement
PASCAL	**37.9**	38.5	34.3	36.4	**4%**	–	43.6	39.8	41.7	**-10%**	**14%**
ImageNet	38.0	**47.9**	40.0	39.0	**19%**	42.9	–	49.1	46.0	**4%**	**18%**
Caltech101	31.9	38.6	**66.6**	35.3	**47%**	34.1	37.4	–	35.8	**46%**	**1%**

beneficial to incorporate the notion of "ease of adaptability" in the process of evaluating datasets — a concept worth further exploring and refining.

9.6 Identifying latent domains from data

We validate our approach to discovering latent domains on visual object recognition and human action recognition tasks. We first describe our evaluation strategy, and then report the results of identifying latent domains and adapting classifiers to a new mono-domain test set. After that, we present experimental results of reshaping heterogeneous test datasets into domains matching to the identified training domains. Finally, we give some qualitative analyses and details on choosing the number of domains.

9.6.1 Evaluation strategy

For object recognition, the four image datasets of Caltech-256 (C), Amazon (A), DSLR (D), and Webcam (W) are commonly used as distinctive domains in research on visual domain adaptation (Saenko et al., 2010; Gopalan et al., 2011; Gong et al., 2012b; Bergamo & Torresani, 2010). Likewise, each view in the IXMAS dataset is often taken as a domain in action recognition (Farhadi & Tabrizi, 2008; Huang et al., 2012; Liu et al., 2011; Li & Zickler, 2012). Similarly, in our experiments, we use a subset of these datasets (views) as source domains for training classifiers and the rest of the datasets (views) as target domains for testing. However, the key difference is that we do not compare performance of different adaptation algorithms which assume domains are already given. Instead, we evaluate the effectiveness of our approach by investigating whether its automatically identified domains improve adaptation, that is, whether recognition accuracy on the target domains can be improved by reshaping the datasets into their latent source domains.

We use the geodesic flow kernel for adapting classifiers (Gong et al., 2012b). To use the kernel-based method for computing distribution difference (*cf.* Chapter 7), we use Gaussian kernels. We set the kernel bandwidth to be twice the median distances of all pairwise data points. The number of latent domains K is determined by the DWCV procedure (*cf.* Section 7.2.2).

9.6.2 Identifying latent domains from training datasets

Notation Let $\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_\mathsf{J}\}$ denote the J datasets we will be using as training source datasets and let $\mathcal{T} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_\mathsf{L}\}$ denote the L datasets we will be using as testing target datasets. Furthermore, let K denote the number of optimal domains discovered by our DWCV procedure and $\mathcal{U} = \{\mathcal{U}_1, \mathcal{U}_2, \ldots, \mathcal{U}_\mathsf{K}\}$ the K hidden domains identified by our approach. Let $r(\mathcal{A} \to \mathcal{B})$ denote the recognition accuracy on the target domain $\mathcal{B}$ with $\mathcal{A}$ as the source domain.

Goodness of the identified domains We examine whether $\{\mathcal{U}_k\}$ is a set of good domains by computing the expected best possible accuracy of using the identified domains *separately* for adaptation

$$G_{\text{OURS}} = \mathbb{E}_{\mathcal{B}\in\mathcal{P}} \max_k r(\mathcal{U}_k, \mathcal{B}) \approx \frac{1}{\mathsf{L}} \sum_l \max_k r(\mathcal{U}_k \to \mathcal{T}_l) \tag{9.1}$$

Table 9.8: Oracle recognition accuracy on target domains by adapting original or identified domains

$\mathcal{S}$	A, C	D, W	C, D, W	Cam 0, 1	Cam 2, 3, 4
$\mathcal{T}$	D, W	A, C	A	Cam 2, 3, 4	Cam 0, 1
G_{ORIG}	41.0	32.6	41.8	44.6	47.1
G_{OTHER} (Hoffman et al., 2012)	39.5	33.7	34.6	43.9	45.1
G_{OURS}	42.6	35.5	44.6	47.3	50.3

Table 9.9: Adaptation recognition accuracies, using original and identified domains with different multi-source adaptation methods

Latent Domains	Multi-DA method	A, C	D, W	C, D, W	Cam 0, 1	Cam 2, 3, 4
		D, W	A, C	A	Cam 2, 3, 4	Cam 0, 1
ORIGINAL	UNION	41.7	35.8	41.0	45.1	47.8
(Hoffman et al., 2012)	ENSEMBLE	31.7	34.4	38.9	43.3	29.6
	MATCHING	39.6	34.0	34.6	43.2	45.2
OURS	ENSEMBLE	38.7	35.8	42.8	45.0	40.5
	MATCHING	42.6	35.5	44.6	47.3	50.3

where $\mathcal{B}$ is a target domain drawn from a distribution on domains $\mathcal{P}$. Since this distribution is not obtainable, we approximate the expectation with the empirical average over the observed testing datasets $\{\mathcal{T}_l\}$. Likewise, we can define G_{ORIG} where we compute the best possible accuracy for the original domains $\{\mathcal{S}_j\}$, and G_{OTHER} where we compute the same quantity for a competing method for identifying latent domains, proposed in (Hoffman et al., 2012). Note that the $\max$ operation requires that the target domains be annotated; thus the accuracies are the most optimistic estimate for all methods, and upper bounds of practical algorithms.

Table 9.8 reports the three quantities on different pairs of sources and target domains. Clearly, our method yields a better set of identified domains, which are always better than the original datasets. We also experimented using K-means or random partition for clustering data instances into domains. Neither yields competitive performance and the results are omitted here for brevity.

Practical utility of identified domains In practical applications of domain adaptation algorithms, however, the target domains are not annotated. The oracle accuracies reported in Table 9.8 are thus not achievable in general. In the following, we examine how closely the performance of the identified domains can approximate the oracle if we employ multi-source adaptation.

To this end, we consider several choices of multiple-source domain adaptation methods:

- UNION The most naive way is to combine all the source domains into a single dataset and adapt from this "mega" domain to the target domains. We use this as a baseline.

- ENSEMBLE A more sophisticated strategy is to adapt each source domain to the target domain and combine the adaptation results in the form of combining multiple classifiers (Hoffman et al., 2012).

Table 9.10: Results of reshaping the test set when it consists of data from multiple domains.

	From identified (Reshaping training only)			No reshaping	Conditional reshaping
	$A' \rightarrow F$	$B' \rightarrow F$	$C' \rightarrow F$	$A \bigcup B \bigcup C \rightarrow F$	$X \rightarrow F_X, \forall X \in \{A', B', C'\}$
Cam 012	36.4	37.1	37.7	37.3	38.5
Cam 123	40.4	38.7	39.6	39.9	41.1
Cam 234	46.5	45.7	46.1	47.8	49.2
Cam 340	50.7	50.6	50.5	52.3	54.9
Cam 401	43.6	41.8	43.9	43.3	44.8

- MATCHING This strategy compares the empirical (marginal) distribution of the source domains and the target domains and selects the *single source domain* that has the smallest difference to the target domain to adapt. We use the kernel-based method to compare distributions, as explained in section 8.3. Note that since we compare only the marginal distributions, we do not require the target domains to be annotated.

Table 9.9 reports the averaged recognition accuracies on the target domains, using either the original datasets/domains or the identified domains as the source domains. The latent domains identified by our method generally perform well, especially using MATCHING to select the single best source domain to match the target domain for adaptation. In fact, contrasting Table 9.9 to Table 9.8, the MATCHING strategy for adaptation is able to match the oracle accuracies, even though the matching process does not use label information from the target domains.

9.6.3 Reshaping the test datasets

So far we have been concentrating on reshaping multiple annotated datasets (for training classifiers) into domains for adapting to test datasets. However, test datasets can also be made of multiple latent domains. Hence, it is also instrumental to investigate whether we can reshape the test datasets into multiple domains to achieve better adaptation results.

However, the reshaping process for test datasets has a critical difference from reshaping training datasets. Specifically, we should reshape test datasets, *conditioning* on the identified domains from the training datasets — the goal is to discover latent domains in the test datasets that match the domains in the training datasets as much as possible. We term this *conditional reshaping*.

Computationally, conditional reshaping is more tractable than identifying latent domains from the training datasets. Concretely, we minimize the distribution differences between the latent domains in the test datasets and the domains in the training datasets, as explained in Section 7.3.

Table 9.10 demonstrates the benefit of conditionally reshaping the test datasets, on cross-view action recognition. This problem inherently needs test set reshaping, since the person may be viewed from any direction at test time. (In contrast, test sets for the object recognition datasets above are less heterogeneous.) The first column shows five groups of training datasets, each being a different view, denoted by A, B and C. In each group, the remaining views D and E are merged into a new test dataset, denoted by $F = D \bigcup E$.

Two baselines are included: (1) adapting from the identified domains A', B' and C' to the merged dataset F; (2) adapting from the merged dataset $A \bigcup B \bigcup C$ to F. These are contrasted to adapting from the identified domains in the training datasets to the *matched* domains in F. In most

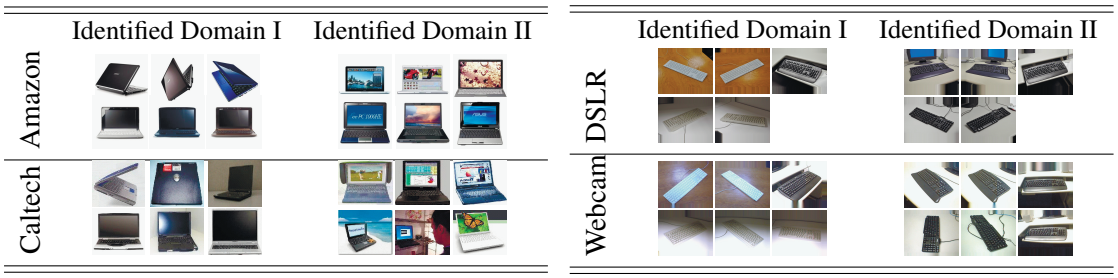

Figure 9.5: Exemplar images from the original and identified domains after reshaping. Note that identified domains contain images from both datasets.

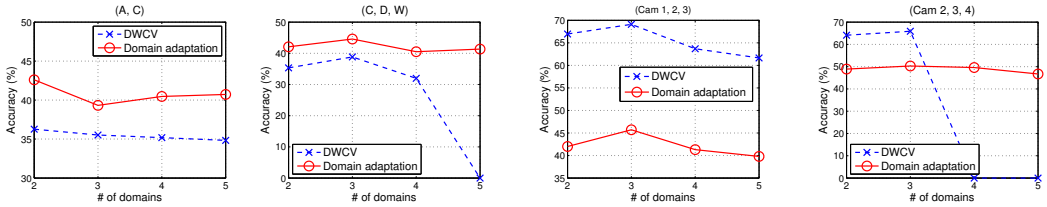

Figure 9.6: Domain-wise cross-validation (DWCV) for choosing the number of domains.

groups, there is a significant improvement in recognition accuracies by conditional reshaping over no reshaping on either training or testing, and reshaping on training only.

9.6.4 Analysis of identified domains and the optimal number of domains

It is also interesting to see which factors are dominant in the identified domains. Object appearance, illumination, or background? Do they coincide with the factors controlled by the dataset collectors?

Some exemplar images are shown in Figure 9.5, where each row corresponds to an original dataset, and each column is an identified domain across two datasets. On the left of Figure 9.5 we reshape Amazon and Caltech-256 into two domains. In Domain II all the "laptop" images 1) are taken from the front view and 2) have colorful screens, while Domain I images are less colorful and have more diversified views. It looks like the domains in Amazon and Caltech-256 are mainly determined by the factors of *object pose* and *appearance (color)*.

The figures on the right are from reshaping DSLR and Webcam, of which the "keyboard" images are taken in an office environment with various lighting, object poses, and background controlled by the dataset creators (Saenko et al., 2010). We can see that the images in Domain II have gray background, while in Domain I the background is either white or wooden. Besides, keyboards of the same model, characterized by color and shape, are almost perfectly assigned to the same domain. In sum, the main factors here are probably *background* and *object appearance (color and shape)*.

Figure 9.6 plots some intermediate results of the domain-wise cross-validation (DWCV) for determining the number of domains K to identify from the multiple training datasets. In addition to the DWCV accuracy $A(\mathsf{K})$, the average classification accuracies on the target domain(s) are also included for reference. We set $A(\mathsf{K})$ to 0 when some categories in a domain are assigned

with only one or no data point (as a result of optimization). Generally, $A(\mathsf{K})$ goes up and then drops at some point, before which is the optimal $\mathsf{K}^\star$ we use in the experiments. Interestingly, the number favored by DWCV coincides with the number of datasets we mix, even though, as our experiments above show, the ideal domain boundaries do not coincide with the dataset boundaries.

9.7 Summary

We have experimentally compared our approaches to several competitive existing methods in the context of sentiment analysis, visual object recognition, and cross-view human action recognition. Our GFK and landmark based adaptation algorithms give rise to the state-of-the-art results on those tasks. In addition to the comparison results, we have also carefully conducted detailed analyses to validate each component in our approaches. The proposed ROD metric is able to rank the source domains according to their adaptabilities to the target domain. The latent domain discovering experiments raise some concerns on the existing benchmark datasets and may shift some research attention in the community to the *domain* itself (in contrast to *adaptation*).

Part V

Conclusion

Chapter 10

Concluding Remarks

We summarize the thesis and discuss some future directions in this section.

10.1 Summary of our work on domain adaptation

This thesis has provided a comprehensive set of techniques with multiple levels of focus on unsupervised domain adaptation, including learning domain-invariant feature representations in order to eliminate the discrepancy between the source domain and the target domain (Chapter 4), manipulating data instances to match the distributions of two domains (Chapter 5), discovering latent domains from heterogeneous data so the individual domains can be better and more efficiently modeled (Chapter 7), and quantizing the "adaptabilities" of different source domains given a particular target domain (Chapter 6).

10.1.1 Domain adaptation algorithms

We develop two feature learning approaches to reducing the gap between the source domain and the target domain, such that the classifiers trained from the source domain also perform well on the target. In the geodesic flow kernel (GFK) (*cf.* Chapter 4), we model the domain shift by the continuously changing subspaces. The overall kernel, GFK, averages out the individual domains' idiosyncrasies. It is computationally efficient thanks to the closed-form solution, conceptually clean with only one free hyper-parameter, and empirically effective as validated on several benchmark datasets for domain adaptation.

In addition to subspaces, we have also studied a new intrinsic structure in domain adaptation, the landmarks (*cf.* Chapter 5). Landmarks are labeled source instances; however, they can also be seen as sampled from the target distribution. Under the covariate shift assumption, the classifiers trained from them are easier to adapt to the target than those trained using the other source instances or the whole source sample. We develop a quadratic programming formulation to automatically identify the landmarks from the source domain. Our GFK and landmark based adaptation methods are among the state-of-the-art algorithms in text sentiment analysis and visual object recognition.

10.1.2 The "adaptability" of a source domain

In terms of how to evaluate the "adaptability" of a source domain without actually running any domain adaptation methods, we propose a rank-of-domains (ROD) metric (*cf.* Chapter 6). ROD accounts for both geometric and statistical properties between two domains. When we have multiple source domains to train our models for the target, we can use the ROD metric to determine the source domain which potentially gives rise to the best performance on the target.

10.1.3 How to define a *domain*

In terms of the notion of *domain*, we raise the concern about the existing benchmark datasets for domain adaptation. Particularly, we claim that it is hard to manually define what constitutes a domain in some applications (*e.g.*, image and video data). If we take the easy alternative and simply treat a dataset as a domain, the adaptation performance is likely limited on the target. To automatically discover domains from data, we propose two axiomatic properties. One is maximum distinctiveness, implying that the domains should be as distinctive from each other as possible. The other is maximum learnability, so that from the discovered domains we can learn strong classifiers. We implement these two properties in a nonparametric fashion and validate our method on visual object recognition and cross-view human action recognition.

10.1.4 Kernel methods in probabilistic models

Our domain adaptation approaches have been taking the advantages of the flexible kernel methods. Finally, we exploit another application of the kernel methods in a probabilistic model, determinantal point process (DPP), and demonstrate its effectiveness in video summarization. We cast video summarization as a supervised subset selection problem, which is in sharp contrast to the existing methods which are largely unsupervised. In particular, we derive a sequential determinantal point process (seqDPP) to learn from human-created video summaries. The seqDPP model encourages the adjacent summary frames diverse from each other and imposes less constraints on the distant frames. It outperforms several competing baselines on three datasets for video summarization.

10.2 Remarks on future work

Domain adaptation is prevalent in various real-world applications. It is in general a fruit-bearing field and is worth more research efforts. In order to develop well-performed algorithms, we have to carefully examine the relationship between domains. However, there could be even no solutions to domain adaptation if the distributions of the source domain and the target domain differ arbitrarily.

We shall continue the research on domain adaptation and hope to develop the next generation of statistical machine learning algorithms which are capable of handling the mismatches in data, *i.e.*, algorithms not limited by the simple assumption that the training and test data are drawn *i.i.d.* from the same distribution. In the following we mainly discuss the future work under two scenarios, video analysis and dealing with massive data.

10.2.1 Structured prediction for temporal video data

Domain adaptation is of great importance for the learning tasks of temporal data, especially video data. About 144,000 hours of video are uploaded to YouTube daily. Around 422,000 CCTV cameras perched around London survey happenings in the city 24/7. With this explosion of video data, there is an ever-pressing need to develop automatic video analysis techniques which often require human labeling information to learn from. However, it is extremely tedious and time-consuming to label the video data — annotators have to, for example, watch a whole video sequence of minutes or hours to create a reliable video summary! On the other hand, we have noticed that there exist many large datasets with labeled images. *Can we alleviate the workload of labeling the temporal video data by adapting the labeling information from the existing datasets of still images?*

We will develop advanced domain adaptation algorithms tailored for the video data to answer this question. Note that structured prediction prevails in video-related tasks, such as video summarization and action localization. However, most existing adaptation methods are neither tuned to model the temporal structure nor designed for structured prediction. Although still one might be able to use them to adapt between the labeled image datasets and the frames of videos, our hunch is that such results will be significantly improved if we can advance the adaptation methods by explicitly modeling the motion cue as a structured prediction problem.

10.2.2 Reducing mismatches in massive data

The emerging "big data" provide great sources of discovery and knowledge. Meanwhile, it poses grand challenges to the current data analysis and management techniques. The National Academies report on "Frontiers in Massive Data Analysis"[1] identifies a list of major challenges that require new approaches to supporting the "big data" era. Among them, **this thesis work is well aligned with "coping with sampling biases and heterogeneity"**, as justified by the following quote from the report.

> "A major issue is that of sampling bias. Data may have been collected according to a certain criterion $\cdots$, but the inferences and decisions $\cdots$ may refer to a different sampling criterion. This issue seems likely to be particularly severe in many massive data sets, which often consist of many subcollections of data, each collected according to a particular choice of sampling criterion and with little control over the overall composition."

My research will explore several avenues to reduce the mismatches in massive data, including: learning good feature representations, developing efficient algorithms to reshape the massive data to domains which are individually more homogeneous, online learning algorithms to update along with the growth of the available (labeled) data, and investigating our models in applied research especially computer vision. The remaining of this section describes some specific future directions.

[1] The report is by the committee on the analysis of massive data, the committee on applied and theoretical statistics, and the board on mathematical sciences and their applications, jointly. It is publicly available at `http://www.nap.edu/catalog/18374/frontiers-in-massive-data-analysis`.

Representation learning Good feature representations should be robust to noises and outliers, invariant to heterogeneity, and concise to save storage and computation resources. Deep learning is powerful and versatile in this sense. Efficient kernel learning and/or approximation methods have also gained some success on large-scale datasets. However, most of them (and other approaches for the same purpose) currently lack the dimension of explicitly modeling the mismatches in data. We believe it is a promising avenue to learn representations which are resilient to the discrepancies in massive data.

Massive data reshaping A heterogeneous large-scale dataset can be regarded as a mixture of several individually more homogeneous sub-collections, or domains. We conjecture that we can gain an overall improvement by reshaping the data to such domains and then modeling them individually, probably with some regularizations similar to multi-task learning. This is partially verified by our empirical findings in (Gong et al., 2013a). Nonetheless, the nonparametric approach developed there is inefficient to handle the large datasets. We anticipate to have some greedy algorithms to implement the maximum distinctiveness and maximum learnability criteria (Gong et al., 2013a).

Online and personalized domain adaptation Online systems are rapidly updating and sophisticating. Almost everyone has multiple online accounts nowadays. As a result, it would be a great convenience for users to have personalized tools to organize and retrieve information. Domain adaptation techniques can facilitate intelligence systems to adapt to different personalized needs. Moreover, online adaptation is highly desirable, considering that the preferences of the users may change over time.

Part VI

Bibliography

Bibliography

Open video project. `http://www.open-video.org/`.

Affandi, R. H., Kulesza, A., and Fox, E. B. Markov determinantal point processes. In *UAI*, 2012.

Aharon, Michal, Elad, Michael, and Bruckstein, Alfred. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

Ando, R.K. and Zhang, T. A framework for learning predictive structures from multiple tasks and unlabeled data. *JMLR*, 6:1817–1853, 2005.

Argyriou, Andreas, Maurer, Andreas, and Pontil, Massimiliano. An algorithm for transfer learning in a heterogeneous environment. In *Machine Learning and Knowledge Discovery in Databases*, pp. 71–85. Springer, 2008.

Bacchiani, Michiel and Roark, Brian. Unsupervised language model adaptation. In *ICASSP*, 2003.

Baktashmotlagh, Mahsa, Harandi, Mehrtash T, Lovell, Brian C, and Salzmann, Mathieu. Unsupervised domain adaptation by domain invariant projection. In *ICCV*, 2013.

Baktashmotlagh, Mahsa, Harandi, Mehrtash, Lovell, Brian, and Salzmann, Mathieu. Domain adaptation on the statistical manifold. In *CVPR*, 2014.

Bay, H., Tuytelaars, T., and Van Gool, L. SURF: Speeded up robust features. In *ECCV*, 2006.

Ben-David, S., Blitzer, J., Crammer, K., and Pereira, F. Analysis of representations for domain adaptation. In *NIPS*, 2007.

Bergamo, A. and Torresani, L. Exploiting weakly-labeled web images to improve object classification: a domain adaptation approach. In *NIPS*, 2010.

Beyer, William H. *CRC standard mathematical tables and formulae*. CRC press, 1991.

Bickel, S., Brückner, M., and Scheffer, T. Discriminative learning for differing training and test distributions. In *ICML*, 2007.

Bickel, Steffen and Scheffer, Tobias. Dirichlet-enhanced spam filtering based on biased samples. In *NIPS*, 2007.

Blitzer, J., McDonald, R., and Pereira, F. Domain adaptation with structural correspondence learning. In *EMNLP*, 2006.

Blitzer, J., Dredze, M., and Pereira, F. Biographies, bolly-wood, boomboxes and blenders: Domain adaptation for sentiment classification. In *ACL*, 2007.

Blitzer, J., Foster, D., and Kakade, S. Domain adaptation with coupled subspaces. In *AISTATS*, 2011.

Bruzzone, L. and Marconcini, M. Domain adaptation problems: A DASVM classification technique and a circular validation strategy. *IEEE PAMI*, 32(5):770–787, 2010.

Cao, L., Liu, Z., and Huang, T. S. Cross-dataset action detection. In *CVPR*, 2010.

Chao, W.-L., Gong, B., Grauman, K., and Sha, F. Large-margin determinantal point processes. In *UAI*, 2015.

Chattopadhyay, Rita, Fan, Wei, Davidson, Ian, Panchanathan, Sethuraman, and Ye, Jieping. Joint transfer and batch-mode active learning. In *ICML*, 2013.

Chelba, C. and Acero, A. Adaptation of maximum entropy capitalizer: Little data can help a lot. *Computer Speech & Language*, 20(4):382–399, 2006.

Chen, Bo, Lam, Wai, Tsang, Ivor, and Wong, Tak-Lam. Extracting discriminative concepts for domain adaptation in text mining. In *KDD*, 2009.

Chen, M., Weinberger, K.Q., and Blitzer, J.C. Co-training for domain adaptation. In *NIPS*, 2011a.

Chen, Minmin, Weinberger, Kilian Q, and Blitzer, John. Co-training for domain adaptation. In *NIPS*, 2011b.

Chen, Minmin, Xu, Zhixiang, Weinberger, Kilian, and Sha, Fei. Marginalized denoising autoencoders for domain adaptation. In *ICML*, 2012.

Chen, Zhiyuan and Liu, Bing. Topic modeling using topics from many domains, lifelong learning and big data. In *ICML*, 2014.

Chopra, Sumit, Balakrishnan, Suhrid, and Gopalan, Raghuraman. Dlid: Deep learning for domain adaptation by interpolating between domains. In *ICML workshop on challenges in representation learning*, 2013.

Ciaramita, Massimiliano and Chapelle, Olivier. Adaptive parameters for entity recognition with perceptron hmms. In *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing*. Association for Computational Linguistics, 2010.

Cortes, Corinna, Mohri, Mehryar, Riley, Michael, and Rostamizadeh, Afshin. Sample selection bias correction theory. In *Algorithmic Learning Theory*, 2008.

Cortes, Corinna, Mansour, Yishay, and Mohri, Mehryar. Learning bounds for importance weighting. In *NIPS*, 2010.

Dai, Wenyuan, Xue, Gui-Rong, Yang, Qiang, and Yu, Yong. Transferring naive bayes classifiers for text classification. In *AAAI*, 2007a.

Dai, Wenyuan, Yang, Qiang, Xue, Gui-Rong, and Yu, Yong. Boosting for transfer learning. In *ICML*, 2007b.

Dai, Wenyuan, Jin, Ou, Xue, Gui-Rong, Yang, Qiang, and Yu, Yong. Eigentransfer: a unified framework for transfer learning. In *ICML*, 2009.

Dang, Hoa Trang. Overview of DUC 2005. In *Document Understanding Conference*, 2005.

Daumé III, H. Frustratingly easy domain adaptation. In *ACL*, 2007.

Daumé III, H. and Marcu, D. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26(1):101–126, 2006.

Daumé III, H., Kumar, A., and Saha, A. Co-regularization based semi-supervised domain adaptation. In *NIPS*, 2010.

Davis, J.V., Kulis, B., Jain, P., Sra, S., and Dhillon, I.S. Information-theoretic metric learning. In *ICML*, 2007.

Dekel, Ofer, Shamir, Ohad, and Xiao, Lin. Learning to classify with missing and corrupted features. *Machine learning*, 81(2):149–178, 2010.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

Dixit, Mandar, Rasiwasia, Nikhil, and Vasconcelos, Nuno. Adapted gaussian models for image classification. In *CVPR*, 2011.

Dollár, P., Wojek, C., Schiele, B., and Perona, P. Pedestrian detection: a benchmark. In *CVPR*, 2009.

Donahue, Jeff, Jia, Yangqing, Vinyals, Oriol, Hoffman, Judy, Zhang, Ning, Tzeng, Eric, and Darrell, Trevor. DeCAF: A deep convolutional activation feature for generic visual recognition. In *ICML*, 2014.

Dredze, Mark, Crammer, Koby, and Pereira, Fernando. Confidence-weighted linear classification. In *ICML*, 2008.

Duan, L., Tsang, IW, Xu, D., and Maybank, SJ. Domain transfer SVM for video concept detection. In *CVPR*, 2009a.

Duan, L., Xu, D., Tsang, I.W., and Luo, J. Visual event recognition in videos by learning from web data. In *CVPR*, 2010.

Duan, Lixin, Tsang, Ivor W, Xu, Dong, and Chua, Tat-Seng. Domain adaptation from multiple sources via auxiliary classifiers. In *ICML*, 2009b.

Duan, Lixin, Tsang, Ivor W, and Xu, Dong. Domain transfer multiple kernel learning. *PAMI*, 34 (3):465–479, 2012a.

Duan, Lixin, Xu, Dong, and Tsang, Ivor. Learning with augmented features for heterogeneous domain adaptation. In *ICML*, 2012b.

Duan, Lixin, Xu, Dong, and Tsang, Ivor W. Domain adaptation from multiple sources: A domain-dependent regularization approach. *IEEE Transactions on Neural Networks and Learning Systems*, 23(3):504–518, 2012c.

Dudík, Miroslav, Phillips, Steven J, and Schapire, Robert E. Correcting sample selection bias in maximum entropy density estimation. In *NIPS*, 2005.

Dumont, Emilie and Mérialdo, Bernard. Automatic evaluation method for rushes summary content. In *ICME*, 2009.

Ellouze, M., Boujemaa, N., and Alimi, A. Im(s)2: Interactive movie summarization system. *J VCIR*, 21(4):283–294, 2010.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The PASCAL Visual Object Classes Challenge 2007, 2007.

Evgeniou, A and Pontil, Massimiliano. Multi-task feature learning. In *NIPS*, 2007.

Fan, Wei, Davidson, Ian, Zadrozny, Bianca, and Yu, Philip S. An improved categorization of classifier's sensitivity on sample selection bias. In *ICDM*, 2005.

Farhadi, A. and Tabrizi, M. Learning to recognize activities from the wrong view point. In *ECCV*, 2008.

Fei-Fei, L., Fergus, R., and Perona, P. Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories. *Computer Vision & Image Understanding*, 106(1):59–70, 2007.

Feng, Shikun, Lei, Zhen, Yi, Dong, and Li, Stan Z. Online content-aware video condensation. In *CVPR*, 2012.

Fernando, Basura, Habrard, Amaury, Sebban, Marc, and Tuytelaars, Tinne. Unsupervised visual domain adaptation using subspace alignment. In *CVPR*, 2013.

Fukumizu, Kenji, Bach, Francis R, and Jordan, Michael I. Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *JMLR*, 5:73–99, 2004.

Fukumizu, Kenji, Gretton, Arthur, Sun, Xiaohai, and Schölkopf, Bernhard. Kernel measures of conditional dependence. In *NIPS*, 2007.

Furini, Marco, Geraci, Filippo, Montangero, Manuela, and Pellegrini, Marco. Stimo: Still and moving video storyboard for the web scenario. *Multimedia Tools and Applications*, 46(1): 47–69, 2010.

Gaidon, Adrien and Vig, Eleonora. Self-learning camera: Autonomous adaptation of object detectors to unlabeled video streams. In *ECCV*, 2014.

Gillenwater, Jennifer, Kulesza, Alex, and Taskar, Ben. Discovering diverse and salient threads in document collections. In *EMNLP/CNLL*, 2012a.

Gillenwater, Jennifer, Kulesza, Alex, and Taskar, Ben. Near-optimal map inference for determinantal point processes. In *NIPS*, 2012b.

Globerson, Amir and Roweis, Sam. Nightmare at test time: robust learning by feature deletion. In *ICML*, 2006.

Glorot, Xavier, Bordes, Antoine, and Bengio, Yoshua. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *ICML*, 2011.

Goldman, Dan B, Curless, Brian, and Seitz, Steven M. Schematic storyboarding for video visualization and editing. In *SIGGRAPH*, 2006.

Gong, B., Sha, F., and Grauman, K. Overcoming dataset bias: An unsupervised domain adaptation approach. In *NIPS Workshop on Large Scale Visual Recognition and Retrieval*, 2012a.

Gong, B., Shi, Y., Sha, F., and Grauman, K. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, 2012b.

Gong, B., Grauman, K., and Sha, F. Reshaping visual datasets for domain adaptation. In *NIPS*, 2013a.

Gong, B., Chao, W.-L., Grauman, K., and Sha, F. Diverse sequential subset selection for supervised video summarization. In *NIPS*, 2014a.

Gong, Boqing, Grauman, Kristen, and Sha, Fei. Connecting the dots with landmarks: discriminatively learning domain-invariant features for unsupervised domain adaptation. In *ICML*, 2013b.

Gong, Boqing, Grauman, Kristen, and Sha, Fei. Learning kernels for unsupervised domain adaptation with applications to visual object recognition. *International Journal of Computer Vision*, 109(1-2):3–27, 2014b.

Gopalan, R., Li, R., and Chellappa, R. Domain adaptation for object recognition: An unsupervised approach. In *ICCV*, 2011.

Gopalan, Raghavan, Li, Ruonan, and Chellappa, Rama. Unsupervised adaptation across domain shifts by generating intermediate data representations. *PAMI*, 36(11):2288–2302, 2014.

Gopalan, Raghuraman. Learning cross-domain information transfer for location recognition and clustering. In *CVPR*, 2013.

Grauman, Kristen and Darrell, Trevor. The pyramid match kernel: Discriminative classification with sets of image features. In *CVPR*, 2005.

Gretton, A., Smola, A., Huang, J., Schmittfull, M., Borgwardt, K., and Scholkopf, B. Covariate shift by kernel mean matching. In Quionero-Candela, J., Sugiyama, M., Schwaighofer, A., and Lawrence, N.D. (eds.), *Dataset Shift in Machine Learning*. MIT Press, 2009.

Gretton, Arthur, Borgwardt, Karsten M, Rasch, Malte, Schölkopf, Bernhard, and Smola, Alex J. A kernel method for the two-sample-problem. In *NIPS*, 2006.

Gretton, Arthur, Sejdinovic, Dino, Strathmann, Heiko, Balakrishnan, Sivaraman, Pontil, Massimiliano, Fukumizu, Kenji, and Sriperumbudur, Bharath K. Optimal kernel choice for large-scale two-sample tests. In *NIPS*, 2012.

Griffin, G., Holub, A., and Perona, P. Caltech-256 object category dataset. Technical report, Caltech, 2007.

Guo, Yuhong and Xiao, Min. Cross language text classification via subspace co-regularized multi-view learning. In *ICML*, 2012.

Habrard, Amaury, Peyrache, Jean-Philippe, and Sebban, Marc. Boosting for unsupervised domain adaptation. In *Machine Learning and Knowledge Discovery in Databases*, pp. 433–448. Springer, 2013.

Hamm, J. and Lee, D.D. Grassmann discriminant analysis: a unifying view on subspace-based learning. In *ICML*, 2008.

Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning*. Springer, 2009.

Heckman, James J. Sample selection bias as a specification error. *Econometrica: Journal of the econometric society*, pp. 153–161, 1979.

Herbrich, Ralf. *Learning kernel classifiers*. MIT Press, Cambridge, 2002.

Hoffman, Judy, Kulis, Brian, Darrell, Trevor, and Saenko, Kate. Discovering latent domains for multisource domain adaptation. In *ECCV*, 2012.

Hoffman, Judy, Rodner, Erik, Darrell, Trevor, Donahue, Jeff, and Saenko, Kate. Efficient learning of domain-invariant image representations. In *ICLR*, 2013.

Hong, R., Tang, J., Tan, H., Yan, S., Ngo, C., and Chua, T. Event driven summarization for web videos. In *Proc. of the first SIGMM Workshop on Social Media*, 2009.

Hou, Xiaodi and Zhang, Liqing. Saliency detection: A spectral residual approach. In *CVPR*, 2007.

Huang, C.-H., Yeh, Y.-R., and Wang, Y.-C. Recognizing actions across cameras by exploring the correlated subspace. In *ECCV*, 2012.

Huang, De-An and Wang, Yu-Chiang Frank. Coupled dictionary and feature space learning with applications to cross-domain image synthesis and recognition. In *ICCV*, 2013.

Huang, Fei and Yates, Alexander. Open-domain semantic role labeling by modeling word spans. In *ACL*, 2010.

Huang, J., Smola, A.J., Gretton, A., Borgwardt, K.M., and Scholkopf, B. Correcting sample selection bias by unlabeled data. In *NIPS*, 2006.

Huang, Jui-Ting and Hasegawa-Johnson, Mark. Maximum mutual information estimation with unlabeled data for phonetic classification. In *InterSpeech*, 2008.

Iyer, Arun, Nath, Saketha, and Sarawagi, Sunita. Maximum mean discrepancy for class ratio estimation: Convergence bounds and kernel selection. In *ICML*, 2014.

Jaakkola, Tommi, Haussler, David, et al. Exploiting generative models in discriminative classifiers. In *NIPS*, 1999.

Jain, V. and Learned-Miller, E. Online domain adaptation of a pre-trained cascade of classifiers. In *CVPR*, 2011.

Jebara, Tony. *Machine learning: discriminative and generative*, volume 755. Springer Science & Business Media, 2012.

Jiang, Jing and Zhai, ChengXiang. Instance weighting for domain adaptation in NLP. In *ACL*, 2007.

Kanamori, Takafumi, Hido, Shohei, and Sugiyama, Masashi. Efficient direct density ratio estimation for non-stationarity adaptation and outlier detection. In *NIPS*, pp. 809–816, 2009a.

Kanamori, Takafumi, Hido, Shohei, and Sugiyama, Masashi. A least-squares approach to direct importance estimation. *JMLR*, 10:1391–1445, 2009b.

Kang, H., Chen, X., Matsushita, Y., and X., Tang. Space-time video montage. In *CVPR*, 2006.

Khosla, A., Hamid, R., Lin, C-J., and Sundaresan, N. Large-scale video summarization using web-image priors. In *CVPR*, 2013.

Kimeldorf, George S and Wahba, Grace. A correspondence between bayesian estimation on stochastic processes and smoothing by splines. *The Annals of Mathematical Statistics*, pp. 495–502, 1970.

Ko, Chun-Wa, Lee, Jon, and Queyranne, Maurice. An exact algorithm for maximum entropy sampling. *Operations Research*, 43(4):684–691, 1995.

Kołcz, Aleksander and Teo, Choon Hui. Feature weighting for improved classifier robustness. In *CEAS'09: sixth conference on email and anti-spam*, 2009.

Kondor, Risi and Jebara, Tony. A kernel between sets of vectors. In *ICML*, pp. 361–368, 2003.

Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012.

Kubat, Miroslav and Matwin, Stan. Addressing the curse of imbalanced training sets: one-sided selection. In *ICML*, 1997.

Kulesza, A. and Taskar, B. Structured determinantal point processes. In *NIPS*, 2011a.

Kulesza, Alex and Taskar, Ben. Learning determinantal point processes. In *UAI*, 2011b.

Kulesza, Alex and Taskar, Ben. k-dpps: Fixed-size determinantal point processes. In *ICML*, 2011c.

Kulesza, Alex and Taskar, Ben. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2-3):123–286, 2012.

Kulis, B., Saenko, K., and Darrell, T. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *CVPR*, 2011.

Kung, Sun Yuan. *Kernel Methods and Machine Learning*. Cambridge University Press, 2014.

Lanckriet, G. R. G., Cristianini, N., Bartlett, P., Ghaoui, L. El, and Jordan, M. Learning the kernel matrix with semidefinite programming. *JMLR*, 5:27–72, 2004.

Lazaric, Alessandro, Brunskill, Emma, et al. Sequential transfer in multi-armed bandit with finite set of models. In *NIPS*, 2013.

Lazebnik, Svetlana, Schmid, Cordelia, and Ponce, Jean. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

Lee, Yong Jae, Ghosh, Joydeep, and Grauman, Kristen. Discovering important people and objects for egocentric video summarization. In *CVPR*, 2012.

Leggetter, Christopher J and Woodland, Philip C. Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models. *Computer Speech & Language*, 9(2):171–185, 1995a.

Leggetter, C.J. and Woodland, P.C. Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. *Computer Speech and Language*, 9(2):171–185, 1995b.

Li, Lianghao, Jin, Xiaoming, and Long, Mingsheng. Topic correlation analysis for cross-domain text classification. In *AAAI*, 2012.

Li, Ruonan and Zickler, Todd. Discriminative virtual views for cross-view action recognition. In *CVPR*, 2012.

Li, Yingbo and Merialdo, Bernard. Vert: automatic evaluation of video summaries. In *Proc. of Multimedia*, 2010.

Liao, Xuejun, Xue, Ya, and Carin, Lawrence. Logistic regression with an auxiliary data source. In *ICML*, 2005.

Lin, Hui and Bilmes, Jeff. Multi-document summarization via budgeted maximization of submodular functions. In *NAACL/HLT*, 2010.

Liu, D., Hua, Gang, and Chen, Tsuhan. A hierarchical visual model for video object summarization. *PAMI*, 32(12):2178–2190, 2010.

Liu, J., Shah, M., Kuipers, B., and Savarese, S. Cross-view action recognition via view knowledge transfer. In *CVPR*, 2011.

Liu, Tiecheng and Kender, John R. Optimization algorithms for the selection of key frame sequences of variable length. In *ECCV*, 2002.

Long, Mingsheng, Ding, Guiguang, Wang, Jianmin, Sun, Jiaguang, Guo, Yuchen, and Yu, Philip S. Transfer sparse coding for robust image representation. In *CVPR*, 2013.

Lowe, David G. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

Lu, Zheng and Grauman, Kristen. Story-driven summarization for egocentric video. In *CVPR*, 2013.

Luo, Jiebo, Papin, Christophe, and Costello, Kathleen. Towards extracting semantically meaningful key frames from personal video clips: from humans to computers. *IEEE Trans. on Circuits and Systems for Video Technology*, 19(2):289–301, 2009.

Ma, Andy J, Yuen, Pong C, and Li, Jiawei. Domain transfer support vector ranking for person re-identification without target camera label information. In *ICCV*, 2013.

Ma, Yu-Fei, Lu, Lie, Zhang, Hong-Jiang, and Li, Mingjing. A user attention model for video summarization. In *ACM Multimedia*, 2002.

Macchi, Odile. The coincidence approach to stochastic point processes. *Advances in Applied Probability*, 7(1):83–122, 1975.

Mansour, Y., Mohri, M., and Rostamizadeh, A. Multiple source adaptation and the rényi divergence. In *UAI*, 2009a.

Mansour, Y., Mohri, M., and Rostamizadeh, A. Domain adaptation with multiple sources. In *NIPS*, 2009b.

Margolis, Anna, Livescu, Karen, and Ostendorf, Mari. Domain adaptation with unlabeled data for dialog act tagging. In *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing*. Association for Computational Linguistics, 2010.

Mohri, Mehryar, Rostamizadeh, Afshin, and Talwalkar, Ameet. *Foundations of machine learning*. MIT press, 2012.

Muandet, Krikamol, Balduzzi, David, and Schölkopf, Bernhard. Domain generalization via invariant feature representation. In *ICML*, 2013.

Müller, Alfred. Integral probability metrics and their generating classes of functions. *Advances in Applied Probability*, pp. 429–443, 1997.

Mundur, Padmavathi, Rao, Yong, and Yesha, Yelena. Keyframe-based video summarization using delaunay clustering. *Int'l J. on Digital Libraries*, 6(2):219–232, 2006.

Ng, Andrew Y and Jordan, Michael I. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *NIPS*, 2002.

Ngo, C.-W., Ma, Y.-F., and Zhang, H.-J. Automatic video summarization by graph modeling. In *ICCV*, 2003.

Ni, Jie, Qiu, Qiang, and Chellappa, Rama. Subspace interpolation via dictionary learning for unsupervised domain adaptation. In *CVPR*, 2013.

Oquab, Maxime, Laptev, Ivan, Bottou, Leon, and Sivic, Josef. Learning and transferring mid-level image representations using convolutional neural networks. In *CVPR*, 2014.

Pan, Sinno Jialin and Yang, Qiang. A survey on transfer learning. *IEEE Trans. Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

Pan, Sinno Jialin, Ni, Xiaochuan, Sun, Jian-Tao, Yang, Qiang, and Chen, Zheng. Cross-domain sentiment classification via spectral feature alignment. In *WWW*, 2010.

Pan, S.J., Tsang, I.W., Kwok, J.T., and Yang, Q. Domain adaptation via transfer component analysis. *IEEE Trans. Neural Nets.*, 99:1–12, 2009.

Pardoe, David and Stone, Peter. Boosting for regression transfer. In *ICML*, pp. 863–870, 2010.

Perronnin, F., Sánchez, J., and Liu, Y. Large-scale image categorization with explicit data embedding. In *CVPR*, 2010.

Perronnin, Florent and Dance, Christopher. Fisher kernels on visual vocabularies for image categorization. In *CVPR*, 2007.

Prettenhofer, Peter and Stein, Benno. Cross-language text classification using structural correspondence learning. In *ACL*, 2010.

Pritch, Y., Rav-Acha, A., Gutman, A., and Peleg, S. Webcam synopsis: Peeking around the world. In *ICCV*, 2007.

Raina, Rajat, Ng, Andrew Y, and Koller, Daphne. Constructing informative priors using transfer learning. In *Proceedings of the 23rd international conference on Machine learning*, 2006.

Ren, Jiangtao, Shi, Xiaoxiao, Fan, Wei, and Philip, S Yu. Type-independent correction of sample selection bias via structural discovery and re-balancing. In *SDM*, 2008.

Reynolds, Douglas A, Quatieri, Thomas F, and Dunn, Robert B. Speaker verification using adapted gaussian mixture models. *Digital signal processing*, 10(1):19–41, 2000.

Rosset, Saharon, Zhu, Ji, Zou, Hui, and Hastie, Trevor J. A method for inferring label sampling mechanisms in semi-supervised learning. In *NIPS*, 2004.

Russell, B. C., Torralba, A., Murphy, K. P., and Freeman, W. T. LabelMe: a database and web-based tool for image annotation. *IJCV*, 77:157–173, 2008.

Saenko, K., Kulis, B., Fritz, M., and Darrell, T. Adapting visual category models to new domains. In *ECCV*, 2010.

Sandra, Eliza Fontes de Avila, Lopes, Ana Paula Brandão, Luz, Antonio da Jr, and Araújo, Arnaldo de Albuquerque. Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method. *Pattern Recognition Letters*, 32(1):56–68, 2011.

Satpal, Sandeepkumar and Sarawagi, Sunita. Domain adaptation of conditional probability models via feature subsetting. In *KDD*, 2007.

Schölkopf, Bernhard and Smola, Alexander J. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

Schölkopf, Bernhard, Smola, Alexander, and Müller, Klaus-Robert. Kernel principal component analysis. In *Artificial Neural NetworksICANN'97*, pp. 583–588. Springer, 1997.

Schölkopf, Bernhard, Herbrich, Ralf, and Smola, Alex J. A generalized representer theorem. In *Computational learning theory*, 2001.

Schölkopf, Bernhard, Janzing, Dominik, Peters, Jonas, Sgouritsa, Eleni, Zhang, Kun, and Mooij, Joris. On causal and anticausal learning. In *ICML*, 2012.

Shawe-Taylor, J. and Cristianini, N. *Kernel methods for pattern analysis*. Cambridge Univ Pr, 2004.

Shekhar, Sumit, Patel, Vishal M., Nguyen, Hien V., and Chellappa, Rama. Generalized domain-adaptive dictionaries. In *CVPR*, June 2013.

Shen, Haoquan, Yu, Shoou-I, Yang, Yi, Meng, Deyu, and Hauptmann, Alex. Unsupervised video adaptation for parsing human motion. In *ECCV*, 2014.

Shi, Lei, Mihalcea, Rada, and Tian, Mingjun. Cross language text classification by model translation and semi-supervised learning. In *ICML*, 2010.

Shi, Yuan and Sha, Fei. Information-theoretical learning of discriminative clusters for unsupervised domain adaptation. In *ICML*, 2012.

Shimodaira, H. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227–244, 2000.

Shrivastava, Ashish, Shekhar, Sumit, and Patel, Vishal M. Unsupervised domain adaptation using parallel transport on grassmann manifold. In *IEEE Winter conference on Applications of Computer Vision*, 2014.

Shu, Le, Ma, Tianyang, and Latecki, Longin Jan. Locality preserving projection for domain adaptation with multi-objective learning. In *AAAI*, 2014.

Smith, Brandon and Zhang, Li. Collaborative facial landmark localization for transferring annotations across datasets. In *ECCV*, 2014.

Smola, Alex, Gretton, Arthur, Song, Le, and Schölkopf, Bernhard. A hilbert space embedding for distributions. In *Algorithmic Learning Theory*, pp. 13–31. Springer, 2007.

Socher, Richard, Ganjoo, Milind, Manning, Christopher D, and Ng, Andrew. Zero-shot learning through cross-modal transfer. In *NIPS*, 2013.

Sriperumbudur, Bharath K, Fukumizu, Kenji, Gretton, Arthur, Lanckriet, Gert RG, and Schölkopf, Bernhard. Kernel choice and classifiability for RKHS embeddings of probability distributions. In *NIPS*, pp. 1750–1758, 2009.

Sriperumbudur, Bharath K, Gretton, Arthur, Fukumizu, Kenji, Schölkopf, Bernhard, and Lanckriet, Gert RG. Hilbert space embeddings and metrics on probability measures. *The Journal of Machine Learning Research*, 99:1517–1561, 2010.

Sriperumbudur, Bharath Kumar. *Reproducing Kernel Space Embeddings and Metrics on Probability Measures*. PhD thesis, University of California, San Diego, 2010.

Steinwart, Ingo. On the influence of the kernel on the consistency of support vector machines. *The Journal of Machine Learning Research*, 2:67–93, 2002a.

Steinwart, Ingo. Support vector machines are universally consistent. *Journal of Complexity*, 18 (3):768–791, 2002b.

Storkey, Amos. When training and test sets are different: characterizing learning transfer. *Dataset shift in machine learning*, pp. 3–28, 2009.

Sugiyama, Masashi and Müller, Klaus-Robert. Input-dependent estimation of generalization error under covariate shift. *Statistics & Decisions*, 23(4/2005):249–279, 2005.

Sugiyama, Masashi, Nakajima, Shinichi, Kashima, Hisashi, Buenau, Paul V, and Kawanabe, Motoaki. Direct importance estimation with model selection and its application to covariate shift adaptation. In *NIPS*, 2008.

Sun, Q., Chattopadhyay, R., Panchanathan, S., and Ye, J. A two-stage weighting framework for multi-source domain adaptation. In *NIPS*, 2011.

Tan, Songbo and Cheng, Xueqi. Improving scl model for sentiment-transfer learning. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pp. 181–184, 2009.

Tang, K., Ramanathan, V., Fei-Fei, L., and Koller, D. Shifting weights: Adapting object detectors from image to video. In *NIPS*, 2012.

Teo, Choon H, Globerson, Amir, Roweis, Sam T, and Smola, Alex J. Convex learning with invariances. In *NIPS*, 2007.

Tommasi, T., Quadrianto, N., Caputo, B., and Lampert, C. Beyond dataset bias: multi-task unaligned shared knowledge transfer. In *ACCV*, 2012.

Tommasi, Tatiana and Caputo, Barbara. Frustratingly easy NBNN domain adaptation. In *ICCV*, 2013.

Torralba, A. and Efros, A.A. Unbiased look at dataset bias. In *CVPR*, 2011.

Tran, D. and Sorokin, A. Human activity recognition with metric learning. In *ECCV*, 2008.

Valdés, Víctor and Martínez, José M. Automatic evaluation of video summaries. *ACM Trans. on multimedia computing, communications, and applications*, 8(3):25, 2012.

Vapnik, Vladimir. *The nature of statistical learning theory*. springer, 2000.

Vella, Francis. Estimating models with sample selection bias: a survey. *Journal of Human Resources*, pp. 127–169, 1998.

Wang, M. and Wang, X. Automatic adaptation of a generic pedestrian detector to a specific traffic scene. In *CVPR*, 2011.

Wang, Xuezhi, Huang, Tzu-kuo, and Schneider, Jeff. Active transfer learning under model shift. In *ICML*, 2014.

Wang, Zheng, Song, Yangqiu, and Zhang, Changshui. Transferred dimensionality reduction. In *Machine learning and knowledge discovery in databases*, pp. 550–565. Springer, 2008.

Weinland, D., Boyer, E., and Ronfard, R. Action recognition from arbitrary views using 3d exemplars. In *ICCV*, 2007.

Wen, Junfeng, Yu, Chun-Nam, and Greiner, Russell. Robust learning under uncertain test distributions: Relating covariate shift to model misspecification. In *ICML*, 2014.

White, Halbert. Consequences and detection of misspecified nonlinear regression models. *Journal of the American Statistical Association*, 76(374):419–433, 1981.

Xu, Zheng, Li, Wen, Niu, Li, and Xu, Dong. Exploiting low-rank structure from latent domains for domain generalization. In *ECCV*, 2014.

Yamada, Makoto, Suzuki, Taiji, Kanamori, Takafumi, Hachiya, Hirotaka, and Sugiyama, Masashi. Relative density-ratio estimation for robust distribution comparison. In *NIPS*, 2011.

Yu, Xiaodong and Aloimonos, Yiannis. Attribute-based transfer learning for object categorization with zero/one training example. In *ECCV*. 2010.

Yu, Yaoliang and Szepesvári, Csaba. Analysis of kernel mean matching under covariate shift. In *ICML*, 2012.

Zadrozny, Bianca. Learning and evaluating classifiers under sample selection bias. In *ICML*, 2004.

Zhang, Hong-Jiang, Wu, Jianhua, Zhong, Di, and Smoliar, Stephen W. An integrated system for content-based video retrieval and browsing. *Pattern Recognition*, 30(4):643–658, 1997.

Zhang, K, Schölkopf, B, Muandet, K, and Wang, Z. Domain adaptation under target and conditional shift. In *ICML*, 2013.

Zheng, Jingjing, Liu, Ming-Yu, Chellappa, Rama, and Phillips, P Jonathon. A grassmann manifold-based domain adaptation approach. In *ICPR*, 2012.

Zheng, Jingjing, Chellappa, Rama, and Phillips, P Jonathon. Sparse embedding-based domain adaptation for object recognition. In *IEEE International Conference on Computer Vision Workshop on Visual Domain Adaptation and Dataset Bias*, 2013.

Part VII

Appendix

A Derivation of the geodesic flow kernel (GFK)

Let $\mathbf{\Omega}^{\mathrm{T}}$ denote the following matrix

$$\mathbf{\Omega}^{\mathrm{T}} = [\boldsymbol{P}_{\mathcal{S}} \;\; \boldsymbol{R}_{\mathcal{S}}] \begin{bmatrix} \boldsymbol{U}_1 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{U}_2 \end{bmatrix}. \tag{1}$$

The geodesic flow $\mathbf{\Phi}(t)$, $t \in (0, 1)$, between $\boldsymbol{P}_{\mathcal{S}}$ and $\boldsymbol{P}_{\mathcal{T}}$ can be written as

$$\mathbf{\Phi}(t) = \boldsymbol{P}_{\mathcal{S}}\boldsymbol{U}_1\mathbf{\Gamma}(t) - \boldsymbol{R}_{\mathcal{S}}\boldsymbol{U}_2\mathbf{\Sigma}(t) = \mathbf{\Omega}^{\mathrm{T}} \begin{bmatrix} \mathbf{\Gamma}(t) \\ -\mathbf{\Sigma}(t) \end{bmatrix}. \tag{2}$$

Recall that the geodesic flow kernel (GFK) is defined as,

$$\langle \boldsymbol{z}_i^{\infty}, \boldsymbol{z}_j^{\infty} \rangle = \int_0^1 (\mathbf{\Phi}(t)^{\mathrm{T}}\boldsymbol{x}_i)^{\mathrm{T}}(\mathbf{\Phi}(t)^{\mathrm{T}}\boldsymbol{x}_j)\, dt = \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{G}\boldsymbol{x}_j, \tag{3}$$

where

$$\boldsymbol{G} = \int_0^1 \mathbf{\Phi}(t)\mathbf{\Phi}(t)^{\mathrm{T}} dt. \tag{4}$$

Substituting the expression of $\mathbf{\Phi}(t)$ of eq. (2) into above, we have (ignoring $\mathbf{\Omega}$ for the moment),

$$\boldsymbol{G} \propto \int_0^1 \begin{bmatrix} \mathbf{\Gamma}(t)\mathbf{\Gamma}(t) & -\mathbf{\Gamma}(t)\mathbf{\Sigma}(t) \\ -\mathbf{\Sigma}(t)\mathbf{\Gamma}(t) & \mathbf{\Sigma}(t)\mathbf{\Sigma}(t) \end{bmatrix} dt \tag{5}$$

Both $\mathbf{\Gamma}(t)$ and $\mathbf{\Sigma}(t)$ are diagonal matrices with elements being $\cos(t\theta_i)$ and $\sin(t\theta_i)$. Thus, we can integrate in close-form,

$$\lambda_{1i} = \int_0^1 \cos^2(t\theta_i) dt = 1 + \frac{\sin(2\theta_i)}{2\theta_i}, \tag{6}$$

$$\lambda_{2i} = -\int_0^1 \cos(t\theta_i)\sin(t\theta_i) dt = \frac{\cos(2\theta_i) - 1}{2\theta_i} \tag{7}$$

$$\lambda_{3i} = \int_0^1 \sin^2(t\theta_i) dt = 1 - \frac{\sin(2\theta_i)}{2\theta_i}, \tag{8}$$

which become the i-th diagonal elements of diagonal matrices $\mathbf{\Lambda}_1$, $\mathbf{\Lambda}_2$, and $\mathbf{\Lambda}_3$ respectively. In terms of these matrices, the inner product eq. (3) is a linear kernel $\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{G}\boldsymbol{x}_j$ with the matrix $\boldsymbol{G}$ given by

$$\boldsymbol{G} = \mathbf{\Omega}^{\mathrm{T}} \begin{bmatrix} \mathbf{\Lambda}_1 & \mathbf{\Lambda}_2 \\ \mathbf{\Lambda}_2 & \mathbf{\Lambda}_3 \end{bmatrix} \mathbf{\Omega}. \tag{9}$$

B Proof of Theorem 1

We first prove the following lemma.

Lemma. *Under the condition of the Theorem 4, the following inequality holds,*

$$KL(P_S\|P_L) \leq KL(P_S\|P_T) \tag{10}$$

Proof We start with

$$\begin{aligned} KL(P_S\|P_T) &= KL(\alpha P_N + (1-\alpha)P_L\|P_T) \\ &= \int [\alpha P_N + (1-\alpha)P_L] \log \frac{\alpha P_N + (1-\alpha)P_L}{P_T}\, dX \end{aligned}$$

We now use the property that $\log$ function is concave to arrive at

$$\begin{aligned} KL(P_S\|P_T) \geq \int [\alpha P_N + (1-\alpha)P_L] \Bigg[& \alpha \log \frac{P_N}{P_T} \\ & + (1-\alpha) \log \frac{P_L}{P_T} \Bigg]\, dX \\ &= \alpha^2 KL(P_N\|P_T) + (1-\alpha)^2 KL(P_L\|P_T) \\ &+ \alpha(1-\alpha) C(P_L, P_N, P_T), \end{aligned} \tag{11}$$

where

$$\begin{aligned} C(P_L, P_N, P_T) &= \int \left(P_N \log \frac{P_L}{P_T} + P_L \log \frac{P_N}{P_T} \right) dX \\ &= \int \left(P_N \log \frac{P_N}{P_T} - P_N \log \frac{P_N}{P_L} + P_L \log \frac{P_L}{P_T} - P_L \log \frac{P_L}{P_N} \right) dX \\ &= KL(P_N\|P_T) - KL(P_N\|P_L) + KL(P_L\|P_T) - KL(P_L\|P_N) \end{aligned} \tag{12}$$

Substituting eq. (12) into eq. (11), we have

$$\begin{aligned} KL(P_S\|P_T) \geq \alpha KL(P_N\|P_T) + (1-\alpha) KL(P_L\|P_T) \\ - \alpha(1-\alpha) \left[KL(P_N\|P_L) + KL(P_L\|P_N) \right] \end{aligned} \tag{13}$$

Applying to the right hand side of the inequality the condition of the Theorem 4, we have

$$KL(P_S\|P_T) \geq \left[\frac{9}{8} - 2\alpha(1-\alpha) \right] A \tag{14}$$

where $A = \max\{KL(P_N\|P_L),\ KL(P_L\|P_N)\}$.

Note that

$$\frac{9}{8} - 2\alpha(1-\alpha) \geq \alpha$$

as the maximum of $2\alpha(1-\alpha) + \alpha$ is $9/8$, attained at $\alpha = 3/4$. This leads to

$$KL(P_S\|P_T) \geq \alpha A \geq \alpha KL(P_N\|P_L) \tag{15}$$

To complete the proof the lemma, note that due to the convexity of KL-divergence, we have

$$KL(P_S\|P_L) = KL(\alpha P_N + (1-\alpha)P_L\|P_L) \leq \alpha KL(P_N\|P_L)$$

Combining the last two inequalities together, we complete the proof of the lemma.

Proof of the Theorem We start by applying the convex property of the KL-divergence again,

$$\begin{aligned} KL(P_S\|Q_T) &= KL(P_S\|\beta P_T + (1-\beta)P_L) \\ &\leq \beta KL(P_S\|P_T) + (1-\beta)KL(P_S\|P_L) \\ &\leq \beta KL(P_S\|P_T) + (1-\beta)KL(P_S\|P_T) \\ &\leq KL(P_S\|P_T) \end{aligned} \tag{16}$$

where we have applied the Lemma 1 in the penultimate inequality. The last inequality states the desired result of the theorem.

C Calculating the softmax of large-margin DPPs

In Section 8.2.3, we use softmax to deal with the exponential number of large-margin constraints and arrive at eq. (8.12) in the main text. Here we show how to calculate the right-hand side of eq. (8.12).

We first compute $\sum_{\boldsymbol{y}\subseteq\mathcal{Y}_n} \ell_\omega(\boldsymbol{y}_n^*, \boldsymbol{y})P(\boldsymbol{y}; \boldsymbol{L}_n)$ as follows

$$\begin{aligned} &\sum_{\boldsymbol{y}\subseteq\mathcal{Y}_n} \ell_\omega(\boldsymbol{y}_n^*, \boldsymbol{y})P(\boldsymbol{y}; \boldsymbol{L}_n) \\ &= \sum_{\boldsymbol{y}\subseteq\mathcal{Y}_n} \left[\sum_{i:i\in\boldsymbol{y}} \mathbb{I}(i \notin \boldsymbol{y}_n^*) + \omega \sum_{i:i\notin\boldsymbol{y}} \mathbb{I}(i \in \boldsymbol{y}_n^*)\right] P(\boldsymbol{y}; \boldsymbol{L}_n) && (17) \\ &= \sum_{i=1}^{\mathsf{M}} \left[\sum_{\boldsymbol{y}:i\in\boldsymbol{y}} \mathbb{I}(i \notin \boldsymbol{y}_n^*)P(\boldsymbol{y}; \boldsymbol{L}_n) + \omega \sum_{\boldsymbol{y}:i\notin\boldsymbol{y}} \mathbb{I}(i \in \boldsymbol{y}_n^*)P(\boldsymbol{y}; \boldsymbol{L}_n)\right] && (18) \\ &= \sum_{i=1}^{\mathsf{M}} \left[\mathbb{I}(i \notin \boldsymbol{y}_n^*)P_{n_{\{i\}}} + \omega\mathbb{I}(i \in \boldsymbol{y}_n^*)\left(1 - P_{n_{\{i\}}}\right)\right] && (19) \\ &= \sum_{i:i\notin\boldsymbol{y}_n^*} P_{n_{\{i\}}} + \omega \sum_{i:i\in\boldsymbol{y}_n^*} (1 - P_{n_{\{i\}}}) && (20) \\ &= \sum_{i:i\notin\boldsymbol{y}_n^*} K_{n_{ii}} + \omega \sum_{i:i\in\boldsymbol{y}_n^*} (1 - K_{n_{ii}}), && (21) \end{aligned}$$

where $P_{n_{\{i\}}} = K_{n_{ii}}$ is the marginal probability of selecting item i. Now we are ready to see

$$\begin{aligned}
&\mathsf{softmax}_{\boldsymbol{y}\subseteq\mathcal{Y}_n} \log \ell_\omega(\boldsymbol{y}_n^*, \boldsymbol{y}) + \log P(\boldsymbol{y}; \boldsymbol{L}_n) \\
&= \log \sum_{\boldsymbol{y}\subseteq\mathcal{Y}_n} \ell_\omega(\boldsymbol{y}_n^*, \boldsymbol{y}) P(\boldsymbol{y}; \boldsymbol{L}_n) \qquad (22) \\
&= \log \left(\sum_{i: i\notin \boldsymbol{y}_n^*} K_{n_{ii}} + \omega \sum_{i: i\in\boldsymbol{y}_n^*} (1 - K_{n_{ii}}) \right). \qquad (23)
\end{aligned}$$

Moreover, recall that $\boldsymbol{K} = \boldsymbol{L}(\boldsymbol{L}+\boldsymbol{I})^{-1}$. Eigen-decomposing $\boldsymbol{L} = \sum_m \lambda_m \boldsymbol{v}_m \boldsymbol{v}_m^T$, we have

$$\begin{aligned}
&\boldsymbol{K} = \boldsymbol{L}(\boldsymbol{L}+\boldsymbol{I})^{-1} = \sum_m \frac{\lambda_m}{\lambda_m + 1} \boldsymbol{v}_m \boldsymbol{v}_m^T, \\
&\text{and thus,} \quad K_{ii} = \sum_m \frac{\lambda_m}{\lambda_m + 1} v_{mi}^2. \qquad (24)
\end{aligned}$$

D Subgradients of the large-margin DPPs

Recall that our objective function in eq. (8.13) of the main text actually consists of a likelihood term $\mathcal{L}(\cdot)$ and the other term of undesirable subsets. Denote them respectively by

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\theta}; \mathcal{Y}_n, \boldsymbol{y}_n^*) &\triangleq \log P(\boldsymbol{y}_n^*; \boldsymbol{L}_n) \\
&= \log\det(\boldsymbol{L}_{n_{\boldsymbol{y}_n^*}}) - \log\det(\boldsymbol{L}_n + \boldsymbol{I}), \qquad (25) \\
\mathcal{A}(\boldsymbol{\theta}; \mathcal{Y}_n, \boldsymbol{y}_n^*) &\triangleq \log \left(\sum_{i\notin\boldsymbol{y}_n^*} K_{n_{ii}} + \omega \sum_{i\in\boldsymbol{y}_n^*} (1 - K_{n_{ii}}) \right). \qquad (26)
\end{aligned}$$

For brevity, we drop the subscript n of $\boldsymbol{L}_n$ and $K_{n_{ii}}$ and change $\boldsymbol{y}_n^*$ to $\boldsymbol{y}_\star$ in what follows.

To compute the overall subgradients, it is sufficient to compute the gradients of the above two terms, $\mathcal{L}$ and $\mathcal{A}$.

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \theta_k} &= \sum_{i,j} \frac{\partial \mathcal{L}}{\partial L_{ij}} \frac{\partial L_{ij}}{\partial \theta_k} = \mathbf{1}^T \left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{L}} \circ \frac{\partial \boldsymbol{L}}{\partial \theta_k} \right) \mathbf{1}, \\
\frac{\partial \mathcal{A}}{\partial \theta_k} &= \mathbf{1}^T \left(\frac{\partial \mathcal{A}}{\partial \boldsymbol{L}} \circ \frac{\partial \boldsymbol{L}}{\partial \theta_k} \right) \mathbf{1}, \qquad (27)
\end{aligned}$$

where $\circ$ stands for the element-wise product between two matrices of the same size. We use the chain rule to decompose $\frac{\partial \boldsymbol{L}}{\partial \theta_k}$ from the overall gradients on purpose. Therefore, if we change the way of parameterizing the DPP kernel $\boldsymbol{L}$, we only need care about $\frac{\partial \boldsymbol{L}}{\partial \theta_k}$ when we compute the gradients for the new parameterization.

D.1 Gradients of the quality-diversity decomposition

In terms of the quality-diversity decomposition (c.f. eq. (7) and (8) in the main text), we have

$$\frac{\partial L_{ij}}{\partial \theta_k} = L_{ij}(x_{ik} + x_{jk}), \qquad \text{or} \quad \frac{\partial \boldsymbol{L}}{\partial \theta_k} = \boldsymbol{L} \circ (\boldsymbol{X}\boldsymbol{e}_k\mathbf{1}^T + \mathbf{1}\boldsymbol{e}_k^T\boldsymbol{X}^T) \tag{28}$$

where $\boldsymbol{q}$ is the vector concatenating the quality terms q_i, $\boldsymbol{X}$ is the design matrix concatenating $\boldsymbol{x}_i^T$ row by row, and $\boldsymbol{e}_k$ stands for the standard unit vector with 1 at the k-th entry and 0 elsewhere.

D.2 Gradients with respect to the DPP kernel

In what follows we calculate $\frac{\partial \mathcal{L}}{\partial \boldsymbol{L}}$ and $\frac{\partial \mathcal{A}}{\partial \boldsymbol{L}}$ in eq. (27). Noting that eq. (27) sums over all the (i, j) pairs, we therefore do not need bother taking special care of the symmetric structure in $\boldsymbol{L}$.

We will need map $\boldsymbol{L}_{\boldsymbol{y}_\star}$ "back" to a matrix $\boldsymbol{M}$ which is the same size as the original matrix $\boldsymbol{L}$, such that $\boldsymbol{M}_{\boldsymbol{y}_\star} = \boldsymbol{L}_{\boldsymbol{y}_\star}$ and all the other entries of $\boldsymbol{M}$ are zeros. We denote by $\langle \boldsymbol{L}_{\boldsymbol{y}_\star} \rangle$ such mapping, i.e., $\langle \boldsymbol{L}_{\boldsymbol{y}_\star} \rangle = \boldsymbol{M}$. Now we are ready to see,

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \boldsymbol{L}} &= \frac{\partial \log\det(\boldsymbol{L}_{\boldsymbol{y}_\star})}{\partial \boldsymbol{L}} - \frac{\partial \log\det(\boldsymbol{L}+\boldsymbol{I})}{\partial \boldsymbol{L}} \\ &= \langle (\boldsymbol{L}_{\boldsymbol{y}_\star})^{-1} \rangle - (\boldsymbol{L}+\boldsymbol{I})^{-1}. \end{aligned} \tag{29}$$

It is a little more involved to compute

$$\frac{\partial \mathcal{A}}{\partial \boldsymbol{L}} = \frac{1}{\sum_{i\notin \boldsymbol{y}_\star} K_{ii} + \omega \sum_{i\in \boldsymbol{y}_\star}(1-K_{ii})} \times \left[\sum_{i\notin \boldsymbol{y}_\star} \frac{\partial K_{ii}}{\partial \boldsymbol{L}} - \omega \sum_{i\in \boldsymbol{y}_\star} \frac{\partial K_{ii}}{\partial \boldsymbol{L}} \right], \tag{30}$$

which involves $\frac{\partial K_{ii}}{\partial \boldsymbol{L}}$.

In order to calculate $\frac{\partial K_{ii}}{\partial \boldsymbol{L}}$, we start from the basic identity (Beyer, 1991) of

$$\frac{\partial \boldsymbol{A}^{-1}}{\partial t} = -\boldsymbol{A}^{-1}\frac{\partial \boldsymbol{A}}{\partial t}\boldsymbol{A}^{-1}, \tag{31}$$

followed by $\frac{\partial \boldsymbol{A}^{-1}}{\partial A_{mn}} = -\boldsymbol{A}^{-1}\boldsymbol{J}^{mn}\boldsymbol{A}^{-1}$, where $\boldsymbol{J}^{mn}$ is the same size as $\boldsymbol{A}$. The (m, n)-th entry of $\boldsymbol{J}^{mn}$ is 1 and all else are zeros.

Let $\boldsymbol{A} = (\boldsymbol{L}+\boldsymbol{I})$. Noting that $\boldsymbol{K} = \boldsymbol{L}(\boldsymbol{L}+\boldsymbol{I})^{-1} = \boldsymbol{I} - (\boldsymbol{L}+\boldsymbol{I})^{-1} = \boldsymbol{I} - \boldsymbol{A}^{-1}$ and thus $K_{ii} = 1 - \left[\boldsymbol{A}^{-1}\right]_{ii}$, we have,

$$\begin{aligned} \frac{\partial K_{ii}}{\partial L_{mn}} &= -\frac{\partial \left[\boldsymbol{A}^{-1}\right]_{ii}}{\partial L_{mn}} = -\frac{\partial \left[\boldsymbol{A}^{-1}\right]_{ii}}{\partial A_{mn}} \\ &= \left[\boldsymbol{A}^{-1}\boldsymbol{J}^{mn}\boldsymbol{A}^{-1}\right]_{ii} = [\boldsymbol{A}^{-1}]_{mi}[\boldsymbol{A}^{-1}]_{ni}. \end{aligned} \tag{32}$$

We can also write eq. (32) in the matrix form,

$$\begin{aligned}\frac{\partial K_{ii}}{\partial \boldsymbol{L}} &= [\boldsymbol{A}^{-1}]_{\cdot i}[\boldsymbol{A}^{-1}]_{\cdot i}^T \\ &= \boldsymbol{A}^{-1}\boldsymbol{e}_i\boldsymbol{e}_i^T\boldsymbol{A}^{-1} = \boldsymbol{A}^{-1}\boldsymbol{J}^{ii}\boldsymbol{A}^{-1},\end{aligned} \tag{33}$$

where $[\boldsymbol{A}^{-1}]_{\cdot i}$ is the i-th column of $\boldsymbol{A}^{-1}$.

Overall, we arrive at a concise form by writing out the right-hand-side of eq. (30) and merging some terms,

$$\begin{aligned}&\sum_{i\notin \boldsymbol{y}_\star} \frac{\partial K_{ii}}{\partial \boldsymbol{L}} - \omega \sum_{i\in \boldsymbol{y}_\star} \frac{\partial K_{ii}}{\partial \boldsymbol{L}} \\ &= \boldsymbol{A}^{-1}\boldsymbol{I}_\omega(\overline{\boldsymbol{y}_\star})\boldsymbol{A}^{-1} = (\boldsymbol{L}+\boldsymbol{I})^{-1}\boldsymbol{I}_\omega(\overline{\boldsymbol{y}_\star})(\boldsymbol{L}+\boldsymbol{I})^{-1}\end{aligned} \tag{34}$$

where $\boldsymbol{I}_\omega(\overline{\boldsymbol{y}_\star})$ looks like an identity matrix except that its (i,i)-th entry is $-\omega$ for $i \in \boldsymbol{y}_\star$.