Publications of the Astronomical Society of the Pacific, 131:108008 (11pp), 2019 October https://doi.org/10.1088/1538-3873/aaf1fc
© 2019. The Astronomical Society of the Pacific. All rights reserved. Printed in the U.S.A.

A General Approach to Domain Adaptation with Applications in Astronomy

Ricardo Vilalta[1], Kinjal Dhar Gupta[1], Dainis Boumber[1], and Mikhail M. Meskhi[2]
[1] Department of Computer Science University of Houston Houston TX, 77204-3010, USA; rvilalta@uh.edu
[2] Department of Computer Science North American University Stafford TX, 77477, USA
Received 2018 August 27; accepted 2018 November 17; published 2019 September 9

Abstract

The ability to build a model on a source task and subsequently adapt this model to a new target task is a pervasive need in many astronomical applications. The problem is generally known in the machine learning field as *transfer learning*, where *domain adaptation* is a popular scenario. An example is to build a predictive model on spectroscopic data to identify Type Ia supernovae (SNe Ia), while subsequently trying to adapt such a model to photometric data. In this paper we propose a new general approach to domain adaptation which does not rely on the proximity of source and target distributions. Instead we simply assume a strong similarity in model complexity across domains, and use active learning to mitigate the dependence on source examples. Our work leads to a new formulation for the likelihood as a function of empirical error using a theoretical learning bound; the result is a novel mapping from generalization error to a likelihood estimation. Results using two real astronomical problems, SN Ia classification and identification of Mars landforms, show two main advantages of our approach: increased performance accuracy and substantial savings in computational cost.

Key words: Domain Adaptation – Supervized Learning – Model Complexity – Supernova Ia – Mars Topography

Online material: color figures

1. Introduction

In this paper we propose a new approach to domain adaptation which is particularly well suited for astronomical applications. Our general setting assumes the learner is embedded in a *domain adaptation* framework (Ben-David et al. 2006, 2010; Blitzer et al. 2006, 2007; Daume & Marcu 2006; Mansour et al. 2009; Hal 2009; Bruzzone & Marconcini 2010), where the goal is to obtain a predictive model on a target domain where examples abound but labeled data are scarce. We assume the existence of a source domain with abundant labeled data, but with a different distribution, such that the naive approach of directly applying the source model onto the target becomes inadequate. Instead we follow a maximum a posteriori (MAP) approach to estimate model complexity by extracting the prior distribution from previous experience (i.e., from a previous task), and by taking the (scarce) target data as evidence to compute the likelihood. The result is a new approach to domain adaptation which is exempt from the common restriction that demands close proximity between source and target distributions (Ben-David et al. 2010).

We show how using a prior distribution from a previous task to estimate a posterior of model complexity on a new task not only yields an increase in performance accuracy, but in addition has an enormous impact on computational cost. Our focus is on astronomical problems where we are witnessing a rapid growth of data volumes corresponding to a variety of astronomical surveys; data repositories have grown from gigabytes to terabytes, and we expect those repositories to reach petabytes in the coming years (Brescia & Longo 2013; Feigelson 2017). Our proposed methodology assumes an exhaustive search for the right model complexity on a source domain, where we generate a prior distribution on model complexity. The arrival of a new target task dispels with such an exhaustive search; instead it generates a posterior distribution that directly leads to finding a near-optimal figure of model complexity. This is particularly important for big-data applications where lengthy computational tasks are unavoidable, even with the availability of an efficient high-performance computing infrastructure.

We report on experiments using two real-world astronomical domains: classification of Type Ia supernovae (SNe Ia) using photometric data, and characterization of landforms on Mars using digital elevation maps (DEMs). Both domains can produce massive amounts of data with a strong need for efficient computational solutions. Results show how the use of a source prior to guide the search for an optimal value of model complexity can significantly improve on generalization performance.

The rationale for assuming similar model complexity values across tasks is based on the nature of distributional discrepancies in many physical domains. The idea is useful not only for astronomical data analysis, but to many other real-world problems where the shift in distribution originates from more sophisticated equipment (e.g., modern telescopes), different

instrumentation, or different coverage on the feature space, while the complexity of the classification problem experiences little change. For example, while spectroscopic and photometric observations capture data at different levels of resolution, the identification itself of specific astronomical objects shares a similar degree of difficulty. In short, we assume that the change in distribution from source to target does not affect model complexity significantly.

This paper is organized as follows. We begin by providing basic concepts in classification and domain adaptation, followed by a detailed description of our proposed approach that shows how to extract a prior distribution from a source domain. We then present our experiments and empirical results. The last section provides a summary and conclusions.

2. Preliminary Concepts

2.1. Basic Notation

In supervised learning or classification, we assume the existence of a training set of examples, $T = \{(\mathbf{x}_i, y_i)\}_{i=1}^{p}$, where the vector $\mathbf{x} = (x_1, x_2, \cdots, x_n)$ is an instance of the input space $\mathcal{X}$, and y is an instance of the output space $\mathcal{Y}$. It is often assumed that sample T contains independently and identically distributed examples that come from a fixed but unknown joint probability distribution, $P(\mathbf{x}, y)$, in the input–output space $\mathcal{X} \times \mathcal{Y}$. The output of the learning algorithm is a function $f_\theta(\mathbf{x})$ (parameterized by θ) mapping the input space to the output space, $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$. Function f_θ comes from a space of functions $\mathcal{H}$. The idea is to search for the hypothesis that minimizes the expectation of a loss function $L(y, f(\mathbf{x}|\theta))$, a.k.a. the risk:

$$R(\theta, P(\mathbf{x}, y)) = E_{\sim P}[L(y, f(\mathbf{x}|\theta))] \quad (1)$$

where we usually employ the zero–one loss function:

$$L(y, f(\mathbf{x}|\theta)) = 1_{\{\mathbf{x}|\mathbf{y}(\mathbf{x}) \neq f(\mathbf{x}|\theta)\}}(\mathbf{x}) \quad (2)$$

such that $1(\cdot)$ is an indicator function, and $\mathbf{y}(\mathbf{x})$ is the true class of $\mathbf{x}$.

Domain Adaptation. In domain adaptation, we assume the existence of a source domain, corresponding to a previous task from which experience can be leveraged, and a target domain, corresponding to the present task. Each domain enables us to draw a data set: $T_s = \{(\mathbf{x}_i, y_i)\}_{i=1}^{p}$ for the source, and $T_t = \{\mathbf{x}_i\}_{i=1}^{q}$ for the target. T_s is an instance of a joint probability distribution, $P_s(\mathbf{x}, y)$, while T_t is an instance of the marginal distribution $P_t(\mathbf{x})$ (from the joint distribution $P_t(\mathbf{x}, y)$, such that $P_t(\mathbf{x}) = \int_y P_t(\mathbf{x}, y) d_y$). Emphasis is always placed on the target domain, corresponding to the task at hand. The main objective is to induce a model from the target data set; when building the model, one can exploit knowledge from the source data set. A major difficulty in domain adaptation stems from the lack of labels on T_t. We will assume the possibility of *querying* some of those examples to attain a few labeled examples as part of an active learning setting (Settles 2012).

Most domain adaptation methods assume similar class posteriors across source and target domains, i.e., $P_s(y|\mathbf{x}) = P_t(y|\mathbf{x})$, but different marginals $P_s(\mathbf{x}) \neq P_t(\mathbf{x})$; this is known as the covariate-shift assumption. Unlike previous work, we will consider the case where both source and target differ in the marginal distributions and class posteriors.

Parameter Estimation. We also consider the problem of parameter estimation, which can play a major role in classification as a means to estimate an optimal figure of model complexity. Examples include finding the number of hidden nodes in a neural network (NN), or finding the degree of a polynomial kernel in support vector machines (SVMs). In MAP, the goal is to obtain a point estimate that maximizes the posterior distribution of the parameter given the data or evidence. The posterior probability is essentially a function of two main factors: the prior probability (i.e., degree of belief of model complexity before data analysis) and the likelihood (i.e., probability of the data sample conditioned on model complexity). When data abound, the likelihood bears a stronger influence on the posterior, while the opposite occurs when data are scarce; here the prior bears more influence on the posterior. An important question is how to obtain a reliable prior when data are scarce (i.e., when the prior plays a strong role in estimating the posterior).

2.2. Related Work

Domain adaptation induces a model by exploiting experience gathered from previous tasks (Ben-David et al. 2010). It is considered a subfield of transfer learning (Pan & Yang 2010), and has become increasingly popular in recent years due to the pervasive nature of task domains exhibiting differences in sample distribution (Liu et al. 2015; Xu et al. 2016). The central question is whether a previously constructed (source) model can be adapted to a new task, or whether it is better to build a new (target) model from scratch.

Domain adaptation methods can be classified into two types: instance-based and feature-based. The idea in instance-based methods is to assign high weights to source examples occupying regions of high density in the target domain. A popular approach is known as covariate shift (Quinonero-Candela et al. 2009; Shimodaira 2000; Kanamori et al. 2009; Sugiyama et al. 2008; Bickel et al. 2009). The covariance-shift assumption is that one can build a model on the newly-weighted source sample and apply it directly to the target domain (Sugiyama et al. 2008; Gretton et al. 2009). A stringent requirement is that source and target distributions must be close to each other.

Feature-based domain adaptation methods attempt to project source and target data sets into a latent feature space, where the covariate-shift assumption holds. A model is then built on the transformed space, and used as the classifier on the target. Examples are structural corresponding learning (Blitzer et al. 2006), subspace alignment methods (Basura et al. 2013),

among others (Ando & Zhang 2005; Glorot et al. 2011; Bruzzone & Marconcini 2010).

From a theoretical view, previous work has tried to estimate the distance between source and target distributions (Ben-David et al. 2006, 2010; Blitzer et al. 2007), and employ regularization terms to find models with good generalization performance on both source and target domains (Kumar et al. 2010).

3. Methodology

We begin by providing a general description of the proposed methodology. The main idea is to assist in finding the right configuration (model complexity) for a learning algorithm by leveraging information from a previous similar task. For example, when trying to find a predictive model to classify SNe, or to predict the class of a transient star, searching for a model with the right degree of complexity by varying a configuration parameter or parameters may turn frustratingly cumbersome. As an example, setting the architecture of a (shallow) NN by varying the number of hidden nodes would lead to a huge number of experiments to assess model quality for each architecture. To alleviate this situation, we *learn* a range of values of model complexity from a previous task using a MAP approach, where there is a high likelihood of finding a good value of model complexity (e.g., number of hidden nodes) on the new task. Moreover, our method disregards many assumptions made by previous work: we do not follow the covariate-shift assumption, no data projection is required to transform the feature space (thus incurring no loss of information), and the dependence on the source is not based on transferring source examples to build the target model. To summarize, the main idea is to learn about the model-building process employed in a previous task (source domain), and to transfer that experience to the new task (target domain). Experience is here understood as a distribution of optimal values of model complexity.

3.1. Active Learning

Our basic strategy is to step aside from the common approach followed by the many domain-adaptation techniques that selectively gather source examples to enlarge the set of target examples. When source and target distributions differ significantly, such an approach can lead to a biased model. Under a high distribution discrepancy, an optimal strategy would simply rely on target instances. But the classical setting in domain adaptation provides no (or very few) class labels on the target data set. A solution to this conundrum is to provide target class labels using active learning (Settles 2012; Balcan et al. 2009; Cohn et al. 1996), where a selective mechanism queries an expert for (target) class labels under a limited budget (i.e., a limited number of queries).

The use of active learning in domain adaptation obviates using source examples while building the target model, opening new research avenues in the field of transfer learning. Here we investigate a mechanism that generates a distribution of model complexity on the source domain, and re-utilizes this distribution as a prior in a Bayesian setting over the target domain. The resulting point-estimate over the posterior distribution of model complexity depends on the prior (source domain) and the likelihood, or evidence (target domain).

3.2. Model Complexity as a Transferable Item

We assume that optimal predictive models for both source and target domains share a similar degree of model complexity. For example, assuming both domains are best modeled using SVMs with a polynomial kernel parameterized by θ (corresponding to the degree of a polynomial), we can then further assume that $P_s(\theta^*) \sim P_t(\theta^*)$, where θ^* is the polynomial degree that minimizes a loss function. This assumption focuses on the similarity of complexity–parameter distributions, and not on the similarity of joint input–output distributions.

Specifically, iteratively sampling and building predictive models on the source domain lead to a distribution of model parameters, $P_s(\theta)$. Our goal is then to estimate an optimal parameter value θ^* that maximizes the posterior distribution on the target domain $P_t(\theta|D)$, where D is the data or evidence (i.e., target sample T_t). By using the distribution gathered from the source domain as a reliable prior, we can formulate the posterior using the Bayes formula:

$$P_t(\theta|D) = \frac{P_t(D|\theta)P_s(\theta)}{\sum_i P_t(D|\theta_i)P_s(\theta_i)} \tag{3}$$

where $P_t(D|\theta)$ is the likelihood, and $P_s(\theta)$ the prior. This is precisely how we propose to adapt a model across domains; assuming the complexity of the model built on the source domain is similar to that on the target domain, we look at the source prior $P_s(\theta)$ as the *transferable item* to be used in the new target domain.

Since the denominator in Equation (3) is constant, we can simplify the equation as follows:

$$P_t(\theta|D) = ZP_t(D|\theta)P_s(\theta) \propto P_t(D|\theta)P_s(\theta) \tag{4}$$

where Z is a normalization factor. To optimize $P_t(\theta|D)$, we optimize the product of $P_t(D|\theta)$ and $P_s(\theta)$, and disregard the value of Z, as it is not a function of parameter θ. Hence, our goal is not to obtain a distribution for the posterior $P_t(\theta|D)$, but only to estimate the value of θ that maximizes the product of the likelihood and prior,[3] the technique known as MAP. We now explain how to compute the prior $P_s(\theta)$ and likelihood

[3] This is different from Bayesian estimation where the output is a full posterior probability distribution over model parameters (Bolstad 2007; Goodman 2005; Louis 2005).

$P_t(D|\theta)$ to obtain a point estimate of model complexity on the target domain.

3.3. Estimating the Likelihood

Our approach to estimate the likelihood is as follows. We first use active learning to obtain a (small) labeled sample from the target domain. We then introduce a novel mechanism to compute $P_t(D|\theta)$ by mapping the generalization error to a likelihood probability. We now explain both steps.

3.3.1. Active Learning

To lessen the dependence on the source domain, we resort to active learning to produce an informative sample of labeled instances from the target domain. We use pool-based active learning with margin sampling (Scheffer et al. 2001) as the uncertainty sampling technique (Lewis & Gale 1994; Lewis & Catlett 1994). Specifically, the algorithm randomly selects an initial set of instances from the unlabeled target data set T_t, and queries their class labels; it then iteratively builds a model $f_t(\mathbf{x}|\theta)$ on the labeled target instances as follows. At every iteration, the algorithm identifies the instance $\mathbf{x}_i$ from the remaining unlabeled target instances with the minimum margin (i.e., minimum distance to the decision boundary), queries $\mathbf{x}_i$ to obtain class label y_i, and adds $(\mathbf{x}_i, y_i)$ to the set of labeled target instances. The process repeats until the budget (i.e., maximum number of allowed queries) is exhausted. The result is a labeled sample that will be used to compute the likelihood $P(D|\theta)$.

3.3.2. Mapping Error to a Likelihood

In general, the likelihood $P(D|\theta)$ estimates the probability of seeing data D given parameter θ. This estimation is particularly complex in our study because θ is normally understood as a parameter of a probabilistic or generative model. $P(D|\theta)$ indicates how likely it is to obtain D from a probabilistic model parameterized by θ. In our case θ is unconventionally defined as a (complexity) parameter of a predictive model $f(x|\theta)$ (e.g., degree of a polynomial kernel).

We contend that $P(D|\theta)$ can be re-interpreted as *the probability of* D given the empirical error of $f(x|\theta)$ on D. In general, the lower the empirical error, the higher the likelihood that model $f(\cdot)$ can reproduce the class labels contained in D.

Our definition of empirical error on D refers to the error incurred on sample D alone, and is denoted as a function of θ and D, $\hat{g}(\theta|D)$. Empirical error is also known as in-sample error.

Our re-interpretation of the likelihood leads naturally to the assumption that the probability of D being classified correctly by a hypothesis $f(\mathbf{x}|\theta)$ is inversely proportional to the error made by $f(\cdot)$ on D. We formulate this inverse relation assuming an exponential distribution:

$$P(D|\theta) = \lambda \exp(-\lambda \hat{g}(\theta|D)) \tag{5}$$

where λ is the rate parameter. This formulation simply states that the likelihood $P(D|\theta)$ decreases exponentially with error $\hat{g}(\theta|D)$, but it is clearly handicapped, as different values of complexity θ are mapped to the same likelihood as long as the empirical error $\hat{g}(\theta|D)$ is identical. However, under equal error rates, we would like to assign a higher likelihood to simpler models. We next propose a solution to this.

3.3.3. Adding Robustness to the New Likelihood

Our formulation of the likelihood as a function of empirical error (Equation (5)) can be made more robust by taking into account the variance component of the error induced by models that belong to families exhibiting high VC-dimension (Vapnik–Chervonenkis dimension; Haussler et al. 1991). In short, we suggest to penalize those scenarios where VC-dimension is high. To begin, we define $g(\theta|D)$ as the expected error across the whole input–output distribution:

$$g(\theta|D) = \int_{\mathcal{X}} \int_{y} L(y, f(\mathbf{x}|\theta)) P(\mathbf{x}, y) dx dy \tag{6}$$

where the loss $L(y, f(\mathbf{x}|\theta)$ is the zero–one loss function. $g(\theta|D)$ is also known as the generalization error. Now, we know from VC inequality (Abu-Mostafa et al. 2012) that, with probability $1 - \delta$, an upper bound on $g(\theta|D)$ is given as follows:

$$g(\theta|D) \leqslant \hat{g}(\theta|D) + \sqrt{\frac{8}{N} \ln \frac{4 m_H(2N)}{\delta}} \tag{7}$$

where δ is user defined, N is the number of training instances, and $m_{\mathcal{H}}(q)$ is a polynomial function that defines that largest number of dichotomies on q training instances given the class of hypotheses $\mathcal{H}$:

$$m_H(q) \leqslant \sum_{i=0}^{d_{\mathrm{VC}}(H)} \frac{N!}{i! N - i!} \tag{8}$$

where $d_{\mathrm{VC}}(H)$ is the VC-dimension (Haussler et al. 1991), defined as the maximum number of examples that can be shattered by $\mathcal{H}$ (Abu-Mostafa et al. 2012), and depends on the complexity of the hypothesis (i.e., on θ). The VC-dimension of various classes of hypotheses is well known. For example, the VC-dimension $d_{\mathrm{VC}}(\mathcal{H})$ of NNs with a sigmoid gate function has a lower bound of $\sigma(w \log w)$ and an upper bound of $O(w^2)$ (Maass 1995), where w is the number of weights in the network. In this example, parameter θ can be interpreted as the number of hidden nodes h in a feed-forward NN. The $d_{\mathrm{VC}}(H)$ of an NN with h hidden nodes can then be estimated using

the lower bound $w \log w$ and defining $w = (i + 1) \times h + (h + 1) \times o$, where i and o are the number of input features and output classes respectively.

We now show how to strengthen our definition of the likelihood. In essence, we keep the exponential distribution the same (Equation (5)), but make parameter λ a function of model complexity θ, $\lambda(\theta)$:

$$P_t(D|\theta) = \lambda(\theta)\exp(-\lambda(\theta)\hat{g}(\theta|D)) \tag{9}$$

and define function $\lambda(\theta)$ as a scaled version of the second part of the VC inequality (Equation (7)):

$$\lambda(\theta) = \alpha\sqrt{\frac{8}{N}\ln\frac{4m_\theta(2N)}{\delta}} \tag{10}$$

where α is a user-defined scale factor that decides how much weight is placed on the variance component of error. By transforming parameter λ into a function parameterized by θ, $\lambda(\theta)$, we achieve our goal of assigning higher likelihoods to simpler models when comparing hypotheses showing similar empirical error. We illustrate these concepts in Figure 1.

The ideas mentioned above have been tried using different strategies. Examples include a full Bayesian approach in transfer learning that finds a common subspace across tasks using a kernel-based dimensionality-reduction technique (Gönen & Margolin 2014), transferring priors across multiple tasks using a hierarchical Bayesian approach (Finkel & Manning 2009), finding clusters of tasks under a Dirichlet process prior (Roy & Kaelbling 2007), finding a Gaussian prior from previous tasks (Raina et al. 2006), and theoretical studies using the PAC learning framework on a Bayesian setting (Germain et al. 2016). All these methods are contingent on the proximity of source and target distributions, whereas our approach relies primarily on the similarity of model complexity.

Embedding the empirical error in an exponential function to compute the likelihood has been tried before (Germain et al. 2016), albeit without considering the capacity of each hypothesis. The novelty of our approach lies in transforming the likelihood function according to the VC-dimension of $\mathcal{H}$.

3.4. Estimating The Prior

Regarding the prior distribution of θ (model complexity) on the source domain, we adopt a parametric model assuming a univariate Gaussian distribution,

$$P_s(\theta) = \frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{\theta-\mu}{\sigma}\right)^2}$$

where μ and σ^2 are the mean and variance respectively. Specifically, our methodology generates k samples of the source data set T_s using uniform random sampling without replacement; k is user-defined, and can be regarded as an experimental design parameter.

We construct classifiers on each of the k samples using a range of model complexity values, $\theta \in \{\theta_1, \theta_2, \theta_3, .. \theta_m\}$. For each sample S_i, we find a value θ_i^*, $1 \leqslant i \leqslant m$, that minimizes the expected loss (i.e., maximizes accuracy). The result is a sample with k optimal values of model complexity. Our estimate of the prior is finally obtained by fitting these values to the Gaussian model.

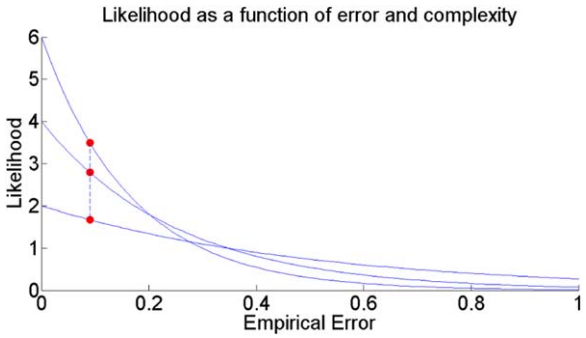

Figure 1. Likelihood of the data given (1) empirical error $\hat{g}(\theta|D)$ and (2) a scaled version of error variance as a function of the VC-dimension $d_{\mathrm{VC}}(\mathcal{H})$. Equal values for $\hat{g}(\theta|D)$ do not map into the same likelihood.
(A color version of this figure is available in the online journal.)

3.5. Estimating the Posterior

Once we have estimated the prior $P_s(\theta)$ and likelihood $P_t(D|\theta)$, we can estimate the numerator of the posterior distribution (Equation (4)): $P_t(\theta|D) = ZP_t(D|\theta)P_s(\theta) \propto P_t(D|\theta)P_s(\theta)$. Since we are interested in obtaining a point estimate for the posterior, we look for an optimal value $\theta^* = \mathrm{argmax}_\theta P_t(D|\theta)P_s(\theta)$.

To reduce the space of complexity values during optimization, we limit the values of θ to the range $[\mu - \sigma, \mu + \sigma]$, where μ and σ are the mean and standard deviation of the source prior distribution $P_s(\theta)$. The final value θ^* is used to build a classifier $f_t(\mathbf{x}|\theta^*)$ on the target domain using the queried instances as our training data set.

Our methodology is outlined in Algorithm 1. The algorithm assumes as input the labeled source data set T_s, the unlabeled target data set T_t, the size r of a small labeled sample to generate a model on the target, and a budget b of possible queries to obtain additional labeled instances on the target. The first step is to build a prior distribution $P_s(\theta) \sim N(\mu, \sigma)$ on the source domain by exhaustively looking for an optimal figure of model complexity; this step can be computationally expensive, but can save substantial amounts of time when it is re-used on a future (target) task. The next steps compute the likelihood $P_t(D|\theta_i)$ in an iterative manner, using labeled instances from the target obtained through active learning. The search is made narrow by limiting values of model complexity to just one standard deviation away from the source mean. The last steps build a (proportional) posterior distribution, and a predictive model using our optimal point estimate for model complexity.

Algorithm 1.

Algorithm 1 Model Complexity Estimation Using Domain Adaptation and Active Learning

Input : Source Dataset T_s, Target Dataset T_t, Budget b, Initial Sample Size r.
Output : Predictive Target Model $f_t^*(\mathbf{x}|\theta)$
1: Estimate prior $P_s(\theta) \sim N(\mu, \sigma)$ using source data set T_s
2: Set $\theta_{\min} = \mu - \sigma$ and $\theta_{\max} = \mu + \sigma$
3: Use the small set of r labeled instances from T_t to build model $f_t(\mathbf{x}|\theta)$
4: Use $f_t(\mathbf{x}|\theta)$ and active learning to label b target instances from T_t
5: **for** $\theta_i \leftarrow \theta_{\min}, \theta_{\max}$
6: Build model $f_t^i(\mathbf{x}|\theta)$ with θ_i as model complexity
7: Compute $\hat{g}(\theta_i|D)$ and $\lambda(\theta_i)$ to estimate likelihood $P_t(D|\theta_i)$
8: Estimate (proportional) posterior: $P_t(D|\theta_i)P_s(\theta_i)$
9: **end for**
10: Let $\theta^* = \text{argmax}_{\theta_i} P_t(D|\theta_i)P_s(\theta_i)$
11: Build $f_t(\mathbf{x}|\theta^*)$
12: **return** $f_t(\mathbf{x}|\theta^*)$

4. Experiments

We next describe our experiments in detail. All our code and data sets have been made available for reproducibility as a github project.[4]

We report empirical results on two different scientific areas to validate our methodology. The first area refers to the automatic classification of SNe using photometric light curves. The second area is centered on the classification of landforms on planet Mars using DEMs.

4.1. Supernova Data Sets

The automatic identification of SNe Ia has become a key step in many astronomical endeavors (Blondin et al. 2012; Sasdelli et al. 2016). Among different types of SNe, SNe Ia are of particular relevance because they can be used as standard candles to probe large cosmological distances. The classification goal here is to identify SNe Ia (positive class) among other types (SNe Ib and Ic, negative class).

When analyzing light from an SN, one can either exploit spectroscopic measurements, to take advantage of the wealth of information that can be obtained from spectral data. This approach, however, is laborious and cumbersome. Another more common approach is to exploit photometric measurements; these are easier to obtain, but limited to a summarization of light intensity in bands or filters. The domain adaptation framework fits into this scenario as follows (Dhar Gupta et al. 2016): the source data set corresponds to spectroscopic measurements where class labels (SNe Ia, Ib, and Ic) are known with high confidence (but data are scarce), whereas the target domain corresponds to photometric measurements where class labels are missing (but data abound).

Our experiments use simulations to generate samples that resemble the type of measurements expected when using spectroscopic or photometric measurements. This brings the advantage of having samples with the same set of features (i.e., same input space $\mathcal{X}$). Specifically, we use data from the Supernova Photometric Classification Challenge (Kessler et al. 2010), consisting of SN light curves simulated according to Dark Energy Survey (DES) specifications, using the SNANA light-curve simulator (Ishida & de Souza 2013). The data come from simulations that approximate the characteristics of the DES. Simulations include both spectroscopic (source) and photometric (target) samples; these are created with biases found in true data sets, where spectroscopic data are in general smaller, brighter, closer, and less noisy than photometric data.

Regarding the construction of simulated samples, we follow the data processing steps specified in Ishida & de Souza (2013). We only take objects with a minimum of three observed epochs per filter; at least one of them occurs before -3 days and at least one after $+24$ days since maximum brightness. On each filter, we use Gaussian process regression to do light-curve model fitting (Chilenski et al. 2015); the resulting function is sampled using a window of size one day. No quality cuts are imposed (signal-to-noise ratio > 0). At the end, the spectroscopic (source) sample has 718 SNe, while the photometric (target) sample has 11,946 SNe. Both samples have 108 columns or features (27 epochs $\times 4$ filters). We use kernel principal component analysis (KPCA) to reduce the original data set from 108 features to only 20 features. This reduction is useful to avoid the curse of dimensionality (Hastie et al. 2001). A preliminary analysis shows no loss of information or performance accuracy after data transformation using KPCA.

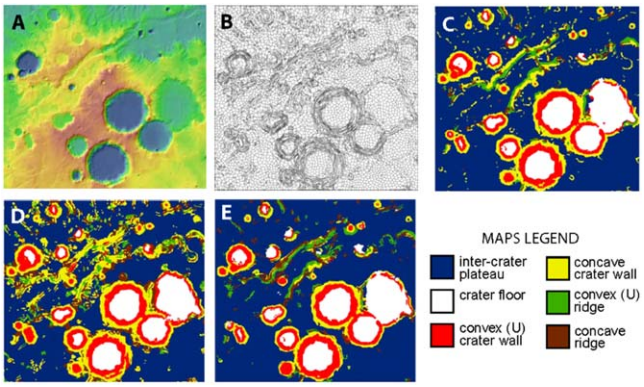

Figure 2. DEMs on Mars processed and classified into different landforms. The original DEM (A), corresponding to a site known as Tissia Valles, is segmented for labeling processes (the red-to-blue gradient indicates high-to-low elevation). (B) The DEM is then classified using models of different complexity (C–E; color labels explained at the bottom right). This site acts as the source domain.
(A color version of this figure is available in the online journal.)

[4] Visit https://github.com/PAL-UH/transferAL.

4.2. Mars Landforms

The second area corresponds to the automatic geomorphic mapping of planetary surfaces. Specifically, the goal is to label segments on specific regions on Mars with their corresponding landforms (Bue & Stepinski 2006; Stepinski et al. 2006; Stepinski & Vilalta 2005). The input data come in the form of raster data or DEMs produced by orbiting satellites (Mars Orbiter Laser Altimeter instrument on board the Mars Global Surveyor spacecraft). Each DEM is first subdivided into meaningful segments (groups of adjacent pixels with similar terrain properties) that are subsequently classified into the following landforms: crater floors, convex crater walls, concave crater walls, convex ridges, concave ridges, and inter-crater plateau. Domain adaptation is important to attain accurate predictive models on new target sites that exhibit different distribution from the original source site.

Figure 2 illustrates the sequence of steps needed to classify landforms on Mars using DEMs. The original map (A) is first divided into small segments amenable to labeling (B). A model is then trained on a fraction of all segments and applied to the rest. Models of different complexity yield different classifications (C–E).

The source domain is a region on Mars where we know the labels for all landforms; it is shown in Figure 2(A) and is known as Tissia Valles; it was chosen primarily because in a relatively small area, most landforms of interest are present. The region is heavily cratered and many different crater morphologies are present in a range of sizes. The goal here is to leverage experience during the model-building process on Tissia Valles to find the right complexity for a model induced on a new site on Mars, corresponding to the target domain, where the landform distribution is different. In our experiments, the target site corresponds to a region known as Evos, shown in Figure 3. Notice the difference in distribution, where the shape of the craters and their number differ significantly from those of Tissia Valles.

The input to the learning algorithm is not the DEM, but a training set made of feature vectors, one for each segment in the map. A segment is made of adjacent pixels with similar feature values. Each segment is characterized by three features, which are averages over the pixels embedded by the segment: slope, computed as the maximum rate of change in elevation from a cell to its neighbors (indicative of the steepness of the terrain); curvature, computed as the second derivative of the surface elevation, useful to distinguish between convex, e.g., ridge, and concave, e.g., channel, surfaces; and flow, computed as the degree of flow accumulated on each cell (high values are indicative of stream or river channels).

4.3. Experimental Settings

To estimate the prior $P_s(\theta)$, we create $k = 100$ bootstrap samples of the source data set T_s under uniform random sampling without replacement. We then record the θ^* that yields highest accuracy on each sample. In our experiments, θ corresponds to the number of hidden nodes in an NN, $\theta \in [2, 50]$. For active learning, we divide data set T_t randomly into two (equal) parts: a pool of target instances from which data can be queried T_t^1, and a pool of test instances that remain unknown during training T_t^2. We then randomly generate 10 pairs of training and testing pools. We first limit our active-learning budget to $b = 100$ queries, and subsequently study how accuracy varies with different budgets.

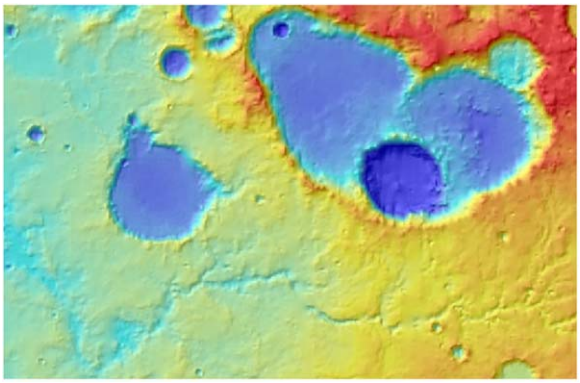

Figure 3. DEM of a region on Mars known as Evos, corresponding to our target domain (the red-to-blue gradient indicates high-to-low elevation). (A color version of this figure is available in the online journal.)

Regarding the posterior, we calculate the value of θ^* that maximizes the product of prior and likelihood. We then use θ^* to build an optimal model $f(\mathbf{x}|\theta^*)$ on the target pool T_t^1, and subsequently test it on the test pool T_t^2. In both domains, we have perfect knowledge of class labels (both source and target samples); these are hidden on the target sample to validate model performance.

Our hardware is a 3712-core computer cluster with eight Tesla C2075 GPUs, 22 GTK570 GPUs, a 120 TB Lustre Filesystem, and 127 TB storage space.

4.4. Methods for Comparison

For comparison purposes, our experiments include other domain adaptation techniques. We next list and describe such techniques.

Subspace Alignment (Basura et al. 2013). The goal is to find separate subspaces for source and target domains using PCA, followed by a linear transformation that maps the source domain into the target domain. The result is an alignment of both spaces through the basis vectors. The number of principal components is optimized based on a theoretical bound.

Joint Distribution Optimal Transportation (JDOT) for domain adaptation (Courty et al. 2017). This technique assumes a map that aligns the joint distributions ($\mathcal{X} \times \mathcal{Y}$) of the source and target domains. The optimization function combines both the distance

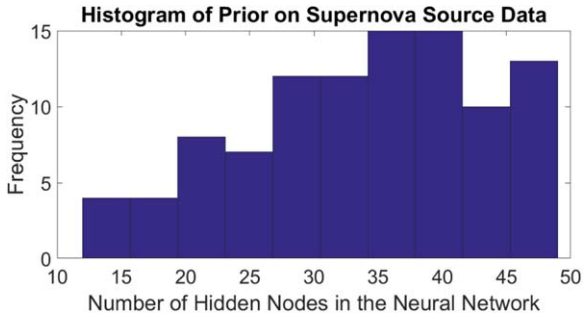

Figure 4. Histogram of optimal values of model complexity for the SN domain.
(A color version of this figure is available in the online journal.)

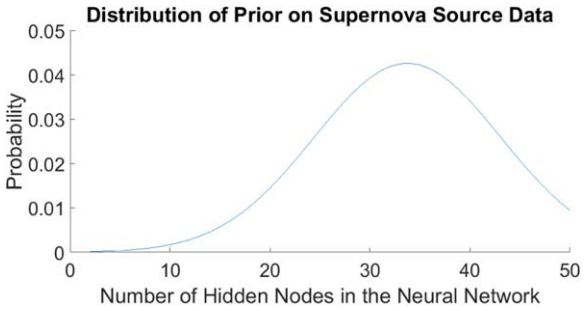

Figure 5. Gaussian approximation to the (prior) distribution of optimal model complexity values for the SN domain.
(A color version of this figure is available in the online journal.)

between samples and the discrepancy in the loss between class labels.

Adaptation Regularization-based Transfer Learning (ARTL; Long et al. 2014). The central idea is to combine different strategies for transfer learning within a single framework: it simultaneously optimizes the structural risk functional over the source domain, the joint distribution matching of both marginal and conditional distributions, and the consistency of the geometric manifold corresponding to the marginal distribution.

Transfer Joint Matching (TJM; Long et al. 2014). This technique combines a shared representation between source and target domains with the concept of instance re-weighting, where source instances that fall within high-density regions of the target domain see their weight increased.

Transfer feature learning with Joint Distribution Adaptation (JDA; Long et al. 2013). This technique jointly adapts both marginal and class-conditional probabilities using PCA.

Domain-Adversarial Training of Neural Networks (DATNN; Ganin et al. 2016). This is a neural network framework that implements domain adaptation by finding features that provide low error on the training set, while the features remain invariant across the two domains (i.e., across source and target domains). The architecture combines two learners that play in an adversarial manner: while one adjusts parameters to reduce training error, the other one adjusts parameters to discriminate (increase error) between source and target examples. The result is a regularized deep neural network that generates an informative abstract feature representation.

Geodesic Flow Kernel (GFK) for unsupervised domain adaptation (Grauman 2012). This technique integrates an infinite number of subspaces between source and target domains by paying attention to geometric and statistical properties of both domains. The technique focuses on those subspaces that are domain invariant.

4.5. Results

Prior. After estimating the sampling distribution for the optimal parameter θ^* on the source domain, our experiments show the following results. For SN: $\mu = 33.75$ and $\sigma = 9.35$, and for Mars landforms: $\mu = 23.19$ and $\sigma = 12.63$. The range of values for the prior are set to $\theta^* \in [24, 43]$ for SN and $\theta^* \in [10, 36]$ for Mars landforms.

Table 1
Classification Accuracy (Numbers Enclosed in Parentheses Represent Standard Deviations)

General Method	Learning Technique	Data Sets	
		Supernova Ia	Mars Landforms
Domain Adaptation	Source Model	69.13 (0.00)	74.36 (9.40)
	Subspace alignment	62.56 (7.98)	85.16 (2.65)
	JDOT SVM	77.57 (0.13)	85.2 (0.59)
	JDOT NN	69.05 (0.08)	80.96 (0.06)
	DATNN	80.4 (0.3)	88.61 (0.22)
	TJM	65.56 (0.01)	82.28 (0.03)
	JDA	70.64 (0.03)	80.40 (0.03)
	ARTL	66.21 (0.01)	88.12 (0.02)
	GFK	63.98 (0.02)	83.56 (0.04)
Source Model + Active Learning	NN + AL	85.75 (0.04)	80.41 (0.08)
	SVM + AL	69.33 (0.17)	85.90 (0.03)
	LR + AL	83.70 (0.03)	85.18 (0.02)
Bayesian DA	NN-DA-AL	86.17 (0.35)	90.81 (1.49)

As an illustration, Figures 4 and 5 show the histogram and corresponding univariate Gaussian approximation of optimal values of model complexity for the SN domain. Model complexity is measured in terms of the number of hidden nodes in a neural network. Approximating the prior distribution helps to narrow the search of values for the posterior, yielding substantial savings in computational cost (as shown below).

Accuracy. Table 1 shows average accuracy comparing our approach with two blocks of techniques: one block labeled "Domain Adaptation" corresponding to the techniques described in Section 4.4 (except for the first technique labeled "Source Model" which simply builds a model on the source domain and applies it directly to the target domain). The other block labeled "Source Model + Active Learning" contains results for methods using active learning, with the initial model

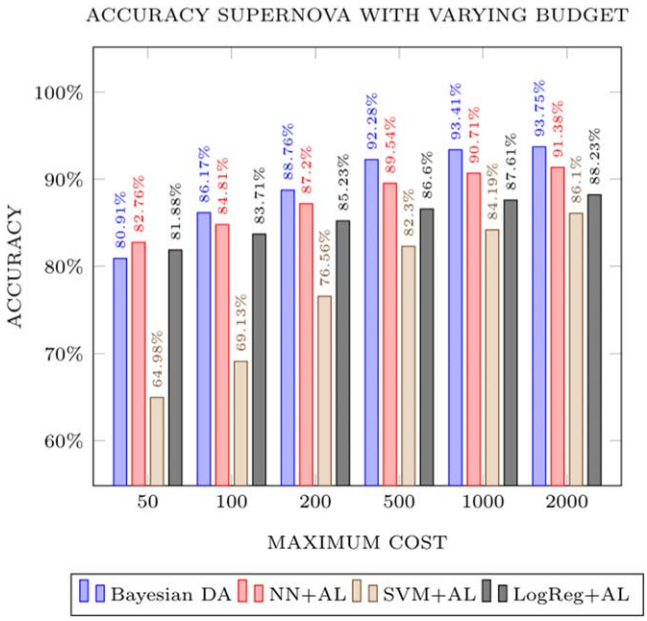

Figure 6. Accuracy on the SN task improves significantly with increasing budget, and tends to converge after about 2000 queries.
(A color version of this figure is available in the online journal.)

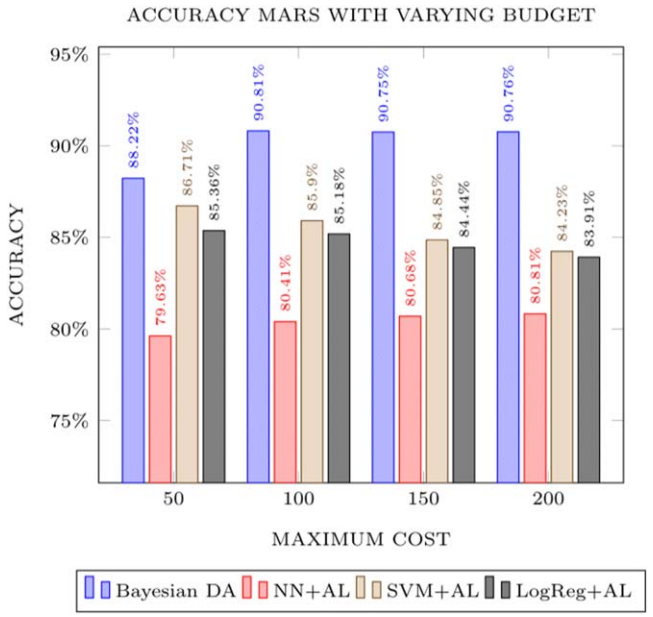

Figure 7. Accuracy on the Mars landforms task improves significantly with a budget of about 100 queries, and tends to converge after that threshold.
(A color version of this figure is available in the online journal.)

built on the source data set. Our technique (Bayesian DA or NN-DA-AL) is shown as the last row of Table 1. It combines domain adaptation with active learning using an NN architecture (budget $b = 100$ queries and the initial labeled pool is of size $r = 10$).

For the first block, results show a significant increase in accuracy with our approach. For a statistical test, we use the Welch's Student paired t-test distribution at the $p = 0.01$ level. We also perform a multiple-comparison test by adjusting the statistical test using a Bonferroni adjustment (Jensen & Cohen 2000); results are significant also at the $p = 0.01$ level after the adjustment, and this is true for both astronomical problems. These results show that using a posterior distribution of model complexity yields better classification accuracy on the target data set than using the best prior. Results also show a major limitation of domain adaptation techniques founded on the assumption of the existence of feature-invariant subspaces between source and target domains. Real-world applications either do not guarantee the existence of such subspaces, or exhibit a complex subspace landscape where finding common subspaces becomes difficult. Under the general assumption where both marginal and posterior probabilities between source and target domains differ ($P_s(\mathbf{x}) \neq P_t(\mathbf{x})$ and $P_s(y|\mathbf{x}) \neq P_t(y|\mathbf{x})$), a better strategy is to sample directly from the target domain under a framework that limits the cost of class queries. If the posterior class probability on the target domain follows a smooth distribution, a limited number of queries should suffice to attain an accurate predictive model.

The second block shows average accuracy with techniques that use active learning, where the initial model is built on the source domain. We report on an NN with logistic activation units and 25 hidden nodes, SVMs with a radial basis function kernel, and logistic regression, all with budget $b = 100$ and $r = 10$. We applied the same statistical test using Welch's Student paired t-test distribution at the $p = 0.01$ and a Bonferroni adjustment. Our approach is significantly more accurate in all cases. This is true even with the use of a plain NN, where there is no search for optimal complexity parameters (e.g., number of hidden units) and no source task to guide such a search. Our method is also more efficient, as finding the best model complexity from scratch on the target domain requires a new and exhaustive search for an optimal value of model complexity.

The next experiment tests the impact of active learning within our approach as the budget is increased. Figure 6 shows results for the SN task and Figure 7 shows results for the Mars landforms task. For the SN task, there is a significant increase in accuracy as the budget grows from 50 to 2000. This is expected as a large labeled sample on the target set provides enough evidence to generate accurate predictive models. Results tend to converge between 500–1000 instances. For the Mars landforms task, results tend to converge after only 100 instances. In practical real-world scenarios, such results can be used to set a trade-off between the size of the budget and the cost of labeling new target instances. In the SN domain, for example, labeling new instances is extremely expensive as it involves running a full spectroscopic analysis; in this case, a

lower budget may be preferred at the cost of some accuracy loss. The opposite is true on the Mars domain, where the cost of labeling segments with their correct landforms is relatively cheap, thus allowing for a budget increase.

Execution Time. A final experiment assesses the benefits gained in computational complexity between our approach and that which finds a common subspace to match source and target distributions. Experiments follow the assumption that domain adaptation generates a prior distribution in the past, and hence does not incur any additional time during the current target task; in addition the search for an optimal value of model complexity on the target is limited to one standard deviation around the optimal value found on the source prior. The model built on the source domain incurs an additional computational cost by searching for a common space over the two domains. We invoke subspace alignment as a representative case of feature matching techniques. For the SN task, execution time is reduced from about 90 hrs to under two hours. For the Mars landforms task, execution time goes from about four hours to about four minutes. Results show the advantage of generating a prior distribution of model complexity on the source domain that is readily available on a new target task: it obviates an exhaustive search for an optimal parameter value.

5. Summary and Conclusions

We propose a new direction in domain adaptation using a MAP approach where the prior distribution is obtained from a source task (previous experience), whereas the likelihood is obtained from the target (or current) task. Our methodology invokes active learning to compensate for the lack of (target) class labels, leaving the budget size as an experimental parameter. Our study leads to a new formulation of the likelihood as a function of empirical error and a term that depends on model complexity as estimated by the VC-dimension. Overall, our technique broadens the general applicability of domain adaptation by relaxing the stringent requirement of close proximity between source and target distributions.

Empirical results on two astronomical problems show a significant advantage in computational cost as the range of complexity values on the target domain is limited to a small window; this is the result of using a prior distribution over the complexity parameter derived from the source domain. In terms of accuracy, results show a significant increase in performance with our approach; this holds for both astronomical domains. Our experiments also show a trade-off between budget size and the cost of labeling; in cases where labeling is relatively cheap, one can increase the budget to achieve an increase in performance accuracy.

As future work, we will investigate how to extend our work when multiple source domains are available. One possibility is to simply choose the best prior based on domain knowledge, or through a ranking system that orders all source tasks based on spatial or temporal proximity to the astronomical event of interest. Another direction is to combine all priors by assigning a degree of relevance to each source task. The posterior distribution can then be defined as a weighted combination of all available priors.

Additionally we hope to stimulate the astronomical community to consider domain adaptation as a useful resource when analyzing different surveys on similar objects. For example, while providing class labels for transient objects or events contained in a single survey is still feasible, although costly, the ability to label variable sources across the large number of available surveys is almost non-existent. The goal of acquiring predictive models from many surveys is a daunting task. This can be tackled by creating predictive models that adapt across the data sets under analysis using domain adaptation techniques. The need for domain adaptation lies in the distributional discrepancy between source and target domains.

This work was partly supported by the Center for Advanced Computing and Data Systems (CACDS), and by the Texas Institute for Measurement, Evaluation, and Statistics (TIMES) at the University of Houston.

References

Abu-Mostafa, Y. S., Magdon-Ismail, M., & Lin, H.-T. 2012, Learning from data, Vol. 4 (Singapore: AMLBook)
Ando, R. K., & Zhang, T. 2005, Journal of Machine Learning Research, 6, 1817 (http://www.jmlr.org/papers/v6/ando05a.html)
Balcan, M.-F., Beygelzimer, A., & Langford, J. 2009, J. Comput. Syst. Sci., 75, 78
Basura, F., Habrard, A., Sebban, M., & Tuytelaars, T. 2013, in Proceedings of the IEEE International Conference on Computer Vision, ICCV, 2960
Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., & Pereira, F. 2010, Mach. Learn., 151
Ben-David, S., Blitzer, J., Crammer, K., & Pereira, F. 2006, in NIPS, ed. B. Schlkopf & J. Platt (Cambridge, MA: MIT Press), 137
Bickel, S., Brückner, M., & Scheffer, T. 2009, J. Mach. Learn. Res., 10, 2137 (http://www.jmlr.org/papers/v10/bickel09a.html)
Blitzer, J., Krammer, K., Kulesza, A., Pereira, F., & Wortman, J. 2007, in Advances in Neural Information Processing Systems, NIPS, 129
Blitzer, J., McDonald, R., & Pereira, F. 2006, in Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, ACL, 120
Blondin, S., Matheson, T., Kirshner, R. P., et al. 2012, AJ, 143, 126
Bolstad, W. 2007, Introduction to Bayesian Statistics (2nd edn; New York: Wiley-Interscience)
Brescia, M., & Longo, G. 2013, NIMPA, 720, 92
Bruzzone, L., & Marconcini, M. 2010, IEEE Trans. Pattern Anal. Mach. Intell., 32, 770
Bue, B., & Stepinski, T. 2006, CG, 32, 604
Chilenski, M., Greenwald, M., Marzouk, Y., et al. 2015, NucFu, 55, 023012
Cohn, D. A., Ghahramani, Z., & Jordan, M. I. 1996, J. Artif. Intell. Res., 4, 129
Courty, N., Flamary, R., Habrard, A., & Rakotomamonjy, A. 2017, in Advances in Neural Information Processing Systems 30, ed. I. Guyon et al. (Red Hook, NY: Curran Associates Inc.), 3730
Daume, H., & Marcu, D. 2006, Journal of Machine Learning Research, 101
Dhar Gupta, K., Pampana, R., Vilalta, R., Ishida, E. E. O., & de Souza, R. S. 2016, Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining, Ser. CIDM '16 (San Francisco, CA: Morgan Kaufmann Publishers Inc.), 61

Feigelson, E. 2017, in IAU Symp. 325, Astroinformatics (Cambridge: Cambridge Univ. Press), 3
Finkel, J. R., & Manning, C. D. 2009, Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Ser. NAACL '09 (Stroudsburg, PA: Association for Computational Linguistics), 602
Ganin, Y., Ustinova, E., Ajakan, H., et al. 2016, Journal of Machine Learning Research, 17, 2096 (http://jmlr.org/papers/v17/15-239.html)
Germain, P., Bach, F., Lacoste, A., & Lacoste-Julien, S. 2016, arXiv:1605.08636
Germain, P., Habrard, A., Laviolette, F., & Morvant, E. 2016, Proceedings of the 33rd International Conference on International Conference on Machine Learning-Volume 48, Ser. ICML'16, 859
Glorot, X., Bordes, A., & Bengio, Y. 2011, Proceedings of the 28th International Conference on Machine Learning (ICML), 513
Gönen, M., & Margolin, A. A. 2014, Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, Ser. AAAI'14 (AAAI Press), 1831
Goodman, S. 2005, Clinical Trials, 2, 281
Grauman, K. 2012, Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Ser. CVPR '12 (Washington, DC: IEEE Computer Society), 2066
Gretton, A., Smola, A., Huang, J., et al. 2009, Dataset Shift in Machine Learning, 3, 5
Hal, D. 2009, arXiv:0907.1815
Hastie, T., Tibshirani, R., & Friedman, J. 2001, The Elements of Statistical Learning, Ser. Springer Series in Statistics. (New York: Springer)
Haussler, D., Kearns, M., & Schapire, R. 1991, Proceedings of the Fourth Annual Workshop on Computational Learning Theory, Ser. COLT '91 (San Francisco, CA: Morgan Kaufmann Publishers Inc.), 61
Ishida, E., & de Souza, R. S. 2013, MNRAS, 430, 509
Jensen, D. D., & Cohen, P. R. 2000, Mach. Learn., 38, 309
Kanamori, T., Hido, S., & Sugiyama, M. 2009, J. Mach. Learn. Res., 10, 1391 (http://www.jmlr.org/papers/v10/kanamori09a.html)
Kessler, R., Conley, A., Jha, S., & Kuhlmann, S. 2010, arXiv:1001.5210
Kumar, A., Saha, A., & Daume, H. 2010, in Advances in Neural Information Processing Systems 23, ed. J. D. Lafferty et al. (Red Hook, NY: Curran Associates, Inc.), 478
Lewis, D. D., & Catlett, J. 1994, in Proceedings of the Eleventh International Conference on Machine Learning, ICML, 148
Lewis, D. D., & Gale, W. A. 1994, in Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 3
Liu, B., Huang, M., Sun, J., & Zhu, X. 2015, Proceedings of the 24th International Conference on Artificial Intelligence, Ser. IJCAI'15 (AAAI Press), 1277
Long, M., Wang, J., Ding, G., Pan, S. J., & Yu, P. S. 2014, IEEE Transactions on Knowledge & Data Engineering, 26, 1076
Long, M., Wang, J., Ding, G., Sun, J., & Yu, P. S. 2013, in The IEEE International Conference on Computer Vision (ICCV)
Long, M., Wang, J., Ding, G., Sun, J., & Yu, P. S. 2014, Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Ser. CVPR '14 (Washington, DC: IEEE Computer Society), 1410
Louis, T. 2005, Clinical Trials, 2, 291
Maass, W. 1995, The Handbook of Brain Theory and Neural Networks, 1000
Mansour, Y., Mohri., M., & Rostamizadeh, A. 2009, in Proceedings of the 22nd Conference on Learning Theory, COLT
Pan, S. J., & Yang, Q. 2010, IEEE Trans. Knowl. Data Eng., 22, 1345
Quinonero-Candela, J., Sugiyama, M., Schwaighofer, A., & Lawrence, N. D. 2009, Dataset shift in Machine Learning (Cambridge, MA: MIT Press)
Raina, R., Ng, A. Y., & Koller, D. 2006, Proceedings of the 23rd International Conference on Machine Learning, Ser. ICML '06 (New York: ACM) 713
Roy, D. M., & Kaelbling, L. P. 2007, Proceedings of the 20th International Joint Conference on Artifical Intelligence, Ser. IJCAI'07 (San Francisco, CA: Morgan Kaufmann Publishers Inc.), 2599
Sasdelli, M., Ishida, E. E. O., Vilalta, R., et al. 2016, MNRAS, 461, 2044
Scheffer, T., Decomain, C., & Wrobel, S. 2001, International Symposium on Intelligent Data Analysis (Berlin: Springer), 309
Settles, B. 2012, Active Learning (New York: Morgan & Claypool)
Shimodaira, H. 2000, J. Stat. Plan. Inference, 90, 227
Stepinski, T. F., Ghosh, S., & Vilalta, R. 2006, in Proceedings of the International Conference on Discovery Science, LNAI, 4265, 255
Stepinski, T., & Vilalta, R. 2005, IEEE Geoscience and Remote Sensing Letters, 2, 260
Sugiyama, M., Nakajima, S., Kashima, H., Buenau, P. V., & Kawanabe, M. 2008, in Advances in Neural Information Processing Systems, NIPS, 1433
Xu, J., Ramos, S., Vázquez, D., & López, A. M. 2016, Int. J. Comput. Vis., 119, 159