Predicting the Future Behavior of a Time-Varying Probability Distribution

Christoph H. Lampert
IST Austria
chl@ist.ac.at

Abstract

We study the problem of predicting the future, though only in the probabilistic sense of estimating a future state of a time-varying probability distribution. This is not only an interesting academic problem, but solving this extrapolation problem also has many practical application, e.g. for training classifiers that have to operate under time-varying conditions.

Our main contribution is a method for predicting the next step of the time-varying distribution from a given sequence of sample sets from earlier time steps. For this we rely on two recent machine learning techniques: embedding probability distributions into a reproducing kernel Hilbert space, and learning operators by vector-valued regression.

We illustrate the working principles and the practical usefulness of our method by experiments on synthetic and real data. We also highlight an exemplary application: training a classifier in a domain adaptation setting without having access to examples from the test time distribution at training time.

1. Introduction

It is a long lasting dream of humanity to build a machine that can predict the future. For long time intervals this is likely going to stay a dream. For shorter time spans, however, this is not such an unreasonable goal. For example, humans can predict rather reliably how a video will continue over the next few seconds. In this work we aim at making a first step towards giving computers similar abilities.

We study the situation of a time-varying probability distribution from which sample sets at different time points are observed. Our main result is a method for learning an operator that captures the dynamics of the time-varying data distribution. It relies on two recent techniques: the embedding of probability distributions into a reproducing kernel Hilbert space, and vector-valued regression. By extrapolating the learned dynamics into the future we obtain an estimate of the future distribution. This estimate can be used to solve practical tasks, for example, learn a classifier that is adapted to the data distribution at a future time step, without having access to data from this situation already. One can also use the estimate to create a new sample set, which then can serve as a drop-in replacement for an actual sample set from the future.

2. Method

We first define the problem setting of *predicting the future of a time-varying probability distribution*. Let $\mathcal{Z}$ be a data domain (formally a *Polish space* [22]), and let $d_t(z)$ for $t \in \mathbb{N}$ be a time-varying data distribution over $z \in \mathcal{Z}$. At a fixed point of time, T, we assume that we have access to sequences of sets, $S_t = \{z_1^t, \dots, z_{n_t}^t\}$, for $t = 1, \dots, T$, that are sampled i.i.d. from the respective distributions, $d_1, \dots, d_T$. Our goal is to construct a distribution, $\tilde{d}_{T+1}$, that is as close as possible to the so far unobserved d_{T+1}, i.e. it provides an estimate of the data distribution one step into the future. Optionally, we are also interested in obtaining a set, $\tilde{S}$, of samples that are distributed approximated according to the unknown d_{T+1}.

Our main contribution is a regression-based method that tackles the above problem for the case when the distribution d_t evolves smoothly (Sections 2.1 and 2.2). We evaluate this method experimentally in Section 4. Subsequently, we show how the ability to extrapolate the distribution dynamics can be exploited to improve the accuracy of a classifier in a domain adaptation setting without observed data from the test time distribution (Section 5).

2.1. Extrapolating the Distribution Dynamics

We propose a method for *extrapolating the distribution dynamics (EDD)* that consists of four steps:

a) represent each sample set as a vector in a Hilbert space,
b) learn an operator that reflects the dynamics between the vectors,
c) apply the operator to the last vector in the sequence, thereby extrapolating the dynamics by one step,
d) (optionally) create a new sample set for the extrapolated distribution.

In the rest of this section we discuss the details of each step, see Figure 1 for a schematic illustration.

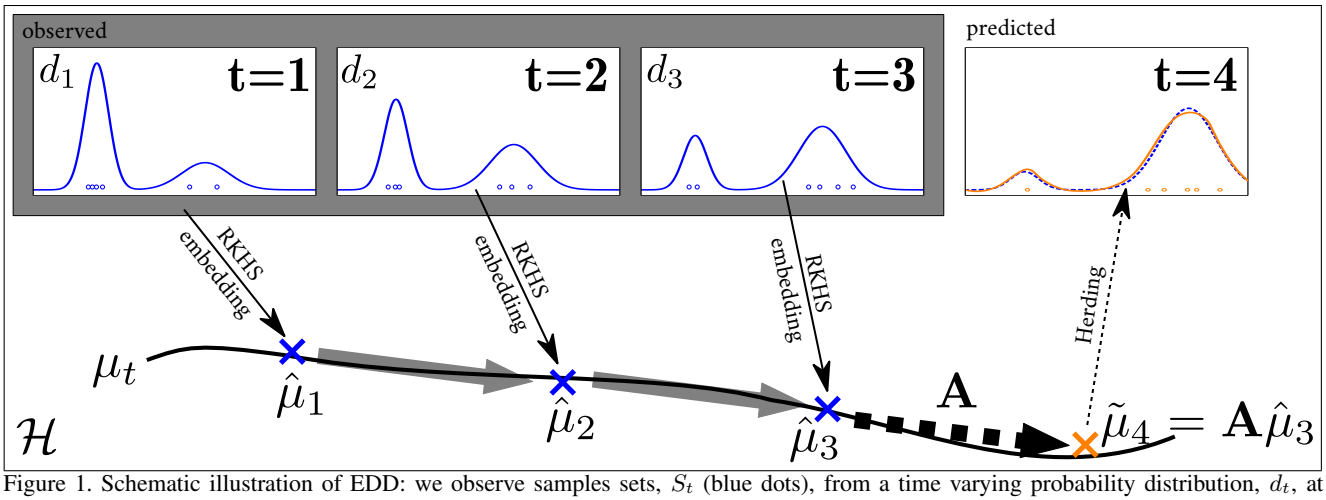

Figure 1. Schematic illustration of EDD: we observe samples sets, S_t (blue dots), from a time varying probability distribution, d_t, at different points of time (blue curves). Using the framework of RKHS embeddings, we compute their empirical kernel mean maps, $\hat{\mu}_t = \frac{1}{|S_t|}\sum_{z\in S_t}\phi(z)$ in a Hilbert space $\mathcal{H}$. We learn an operator $\mathbf{A} : \mathcal{H} \to \mathcal{H}$ that approximates the dynamics from any $\hat{\mu}_t$ to $\hat{\mu}_{t+1}$ by vector-valued regression (thick gray arrows). By means of $\mathbf{A}$ we extrapolate the distribution dynamics beyond the last observed distribution (thick dashed arrow), thereby obtaining a prediction, $\tilde{\mu}_4$, for the embedding of the unobserved *target* distribution d_4 (dotted blue curve). If desired, we apply *herding* (thin dashed arrow) to produce a new sample set (orange dots) for the predicted distribution (orange curve).

a) RKHS Embedding. In order to allow the handling of arbitrary real data, we would like to avoid making any domain-specific assumptions, such as that the samples correspond to objects in a video, or parametric assumptions, such as Gaussianity of the underlying distributions. We achieve this by working in the framework of *reproducing kernel Hilbert space (RKHS) embeddings of probability distributions* [19]. In this section we provide the most important definitions; for a comprehensive introduction see [20].

Let $\mathcal{P}$ denote the set of all probability measures on the data domain $\mathcal{Z}$. Let $k : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$ be a positive definite kernel function with induced RKHS $\mathcal{H}$ and feature map $\phi : \mathcal{Z} \to \mathcal{H}$ that fulfills $\|\phi(z)\| \leq 1$ for all $z \in \mathcal{Z}$. The *kernel mean embedding*, $\mu : \mathcal{P} \to \mathcal{H}$, associated with k is defined by

$$p \mapsto \mu(p), \quad \text{for } \mu(p) = \mathbb{E}_{z\sim p(z)}\{\phi(z)\}. \tag{1}$$

Since we assume k (and therefore $\mathcal{H}$) fixed in this work, we also refer to $\mu(p)$ as "the" RKHS embedding of p. We denote by $\mu(\mathcal{P})$ the image of $\mathcal{P}$ under μ, i.e. the set of vectors that correspond to embedded probability distributions. For *characteristic* kernels, such as the Gaussian, the kernel mean map is an *bijection* between $\mathcal{P}$ and $\mu(\mathcal{P})$, so no information is lost by the embedding operation [19]. In the rest of this section, we will use the term distribution to refer to objects either in $\mathcal{P}$ or in $\mu(\mathcal{P})$, when it is clear from the context which ones we mean.

A noteworthy property of the kernel mean map is that it allows us to express the operation of taking expected values by an inner product using the identity $\mathbb{E}_{z\sim p(z)}\{f(z)\} = \langle \mu(p), f\rangle_{\mathcal{H}}$ for any $p \in \mathcal{P}$ and $f \in \mathcal{H}$.

For a set $S = \{z_1, \ldots, z_n\}$ of i.i.d. samples from p,

$$S \mapsto \hat{\mu}(S), \quad \text{for } \hat{\mu}(S) = \frac{1}{n}\sum_{z\in S}\phi(z) \tag{2}$$

is called the *empirical (kernel mean) embedding* of S. It is known that under mild conditions on $\mathcal{H}$, the empirical embedding, $\hat{\mu}(S)$, converges with high probability to the true embedding, $\mu(p)$, at a rate of $O(1/\sqrt{n})$ [1].

The first step of EDD consists of forming the embeddings, $\hat{\mu}_1, \ldots, \hat{\mu}_T$ of the observed sample sets, $S_1, \ldots, S_T$. Note that for many interesting kernels the vectors $\hat{\mu}_t$ cannot be computed explicitly, because the kernel feature map, ϕ, is unknown or would require infinite memory to be represented. However, as we will see later and as it is typical for kernel methods [13], explicit knowledge of the embedding vectors is also not required. It is sufficient that we are able to compute their inner products with other vectors, and this can be done via evaluations of the kernel function.

b) Learning the Dynamics. We use *vector-valued regression* [16] to learn a model of the process how the (embedded) distribution evolves from one time step to the next. Vector-valued regression generalizes classical scalar-valued regression to the situation in which the inputs and outputs are vectors, i.e. the learning of an operator. Again, we start by providing a summary of this technique, here following the description in [15].

As basis set in which we search for a suitable operator, we define a space, $\mathcal{F}$, of linear operators on $\mathcal{H}$ in the following way. Let $\mathcal{L}(\mathcal{H})$ be the space of all bounded linear operators from $\mathcal{H}$ to $\mathcal{H}$, and let $L : \mathcal{H} \times \mathcal{H} \to \mathcal{L}(\mathcal{H})$ be

the *nonnegative $\mathcal{L}(\mathcal{H})$-valued kernel* defined by $L(f,g) = \langle f,g\rangle_{\mathcal{H}} \operatorname{Id}_{\mathcal{H}}$ for any $f,g \in \mathcal{H}$, where $\operatorname{Id}_{\mathcal{H}}$ is the identity operator on $\mathcal{H}$. Then L can be shown to be the reproducing kernel of an operator-valued RKHS, $\mathcal{F} \subseteq \mathcal{L}(\mathcal{H})$, which contains at least the span of all rank-1 operators, $fg^* : \mathcal{H} \to \mathcal{H}$, for all $f,g \in \mathcal{H}$, with $g^*(\cdot) = \langle g, \cdot\rangle_{\mathcal{H}}$. The inner product between such operators is $\langle f_1 g_1^*, f_2 g_2^*\rangle_{\mathcal{F}} = \langle f_1, g_1\rangle_{\mathcal{H}}\langle f_2, g_2\rangle_{\mathcal{H}}$ for any $f_1, g_1, f_2, g_2 \in \mathcal{H}$, and this induces the inner product of all other operators in $\mathcal{F}$ by linearity and completeness.

As second step of EDD we solve a vector-valued regression in order to learn a predictive model of the dynamics of the distribution. For this we assume that the changes of the distributions between time steps can be approximated by an autoregressive process, e.g. $\mu_{t+1} = \mathbf{A}\mu_t + \epsilon_t$, for some operator $\mathbf{A} : \mathcal{H} \to \mathcal{H}$, such that the ϵ_t for $t = 1,\ldots,T$ are independent zero-mean random variables. To learn the operator we solve the following least-squares functional with regularization constant $\lambda \geq 0$:

$$\min_{A\in\mathcal{F}} \quad \sum_{t=1}^{T-1} \|\hat{\mu}_{t+1} - A\hat{\mu}_t\|_{\mathcal{H}}^2 \;+\; \lambda\|A\|_{\mathcal{F}}^2. \tag{3}$$

Equation (3) has a closed-form solution,

$$\tilde{\mathbf{A}} = \sum_{t=1}^{T-1} \hat{\mu}_{t+1} \sum_{s=1}^{T-1} W_{ts}\hat{\mu}_s^*, \tag{4}$$

with coefficient matrix $W = (K + \lambda I)^{-1}$, where $K \in \mathbb{R}^{(T-1)\times(T-1)}$ is the kernel matrix with entries $K_{st} = \langle \hat{\mu}_s, \hat{\mu}_t\rangle_{\mathcal{H}}$, and I is the identity matrix of the same size, see [15] for the derivation. Recently, it has been shown that the above regression on distributions is consistent under certain technical conditions [23]. Consequently, if $\mathbf{A} \in \mathcal{F}$, then the estimated operator, $\tilde{\mathbf{A}}$, will converge to the true operator, $\mathbf{A}$, when the number of sample sets and the number of samples per set tend to infinity.

c) Extrapolating the Evolution. The third step of EDD is to extrapolate the dynamics of the distribution by one time step. With the results of a) and b), all necessary components for this are available: we simply apply the learned operator, $\tilde{\mathbf{A}}$ to the last observed distribution $\hat{\mu}_T$. The result is a prediction, $\tilde{\mu}_{T+1} = \tilde{\mathbf{A}}\hat{\mu}_T$, that approximates the unknown target, μ_{T+1}. From Equation (4) we see that $\tilde{\mu}_{T+1}$ can be written as a weighted linear combination of the observed distributions,

$$\tilde{\mu}_{T+1} = \sum_{t=2}^{T} \beta_t \hat{\mu}_t \;\text{ with }\; \beta_{t+1} = \sum_{s=1}^{T-1} W_{ts}\langle \hat{\mu}_s, \hat{\mu}_T\rangle_{\mathcal{H}}, \tag{5}$$

for $t = 1,\ldots,T-1$. The coefficients, β_t, can be computed from the original sample sets by means of only kernel evaluations, because $\langle \hat{\mu}_s, \hat{\mu}_t\rangle_{\mathcal{H}} = \frac{1}{n_s n_t}\sum_{i=1}^{n_s}\sum_{j=1}^{n_t} k(z_i^s, z_j^t)$. Their values can be positive or negative, so $\tilde{\mu}_{T+1}$ is not just an interpolation between previous values but potentially an extrapolation. In particular it can lie outside of the convex hull of the observed distributions. At the same time, the estimate $\tilde{\mu}_{T+1}$ is guaranteed to lie in the subspace spanned by $\hat{\mu}_2,\ldots,\hat{\mu}_T$, for which we have sample sets available. Therefore, so we can compute expected values with respect to $\tilde{\mu}_{T+1}$ by forming a suitably weighted linear combinations of the target function at the original data points. For any $f \in \mathcal{H}$, we have

$$\begin{aligned}\tilde{\mathbb{E}}_{\tilde{\mu}_{T+1}}\{f\} = \langle \tilde{\mu}_{T+1}, f\rangle_{\mathcal{H}} &= \sum_{t=2}^{T} \beta_t \big\langle \hat{\mu}_t, f\big\rangle_{\mathcal{H}} \\ &= \sum_{t=2}^{T} \beta_t \frac{1}{n_t}\sum_{i=1}^{n_t} \big\langle \phi(z_i^t), f\big\rangle_{\mathcal{H}} = \sum_{t=2}^{T}\sum_{i=1}^{n_t} \frac{\beta_t}{n_t} f(z_i^t),\end{aligned} \tag{6}$$

where the last identity is due to the fact that $\mathcal{H}$ is the RKHS of k, which has ϕ as its feature map, so $\langle \phi(z), f\rangle_{\mathcal{H}} = f(z)$ for all $z \in \mathcal{Z}$ and $f \in \mathcal{H}$. We use the symbol $\tilde{\mathbb{E}}$ instead of $\mathbb{E}$ to indicate that $\langle \tilde{\mu}_{T+1}, f\rangle_{\mathcal{H}}$ does not necessarily correspond to the operation of computing an expected value, because $\tilde{\mu}_{T+1}$ might not have a pre-image in the space of probability distributions. The following lemma shows that $\tilde{\mu}_{T+1}$ can, nevertheless, act as a reliable proxy for μ_{T+1}:

Lemma 1. *Let $\mu_{T+1} = \mathbf{A}\mu_T + \epsilon_T$ and $\tilde{\mu}_{T+1} = \tilde{\mathbf{A}}\hat{\mu}_T$, for some $\mu_T \in \mu(\mathcal{P})$, $\hat{\mu}_T, \epsilon_T \in \mathcal{H}$ and $\mathbf{A}, \tilde{\mathbf{A}} \in \mathcal{F}$. Then the following inequality holds for all $f \in \mathcal{H}$ with $\|f\|_{\mathcal{H}} \leq 1$,*

$$\begin{aligned}|\mathbb{E}_{\mu_{T+1}}\{f\} - \tilde{\mathbb{E}}_{\tilde{\mu}_{T+1}}\{f\}| \leq\; &\|\mathbf{A}\|_{\mathcal{F}}\|\mu_T - \hat{\mu}_T\|_{\mathcal{H}} \\ &+ \|\mathbf{A} - \tilde{\mathbf{A}}\|_{\mathcal{F}} \;+\; \|\epsilon_T\|_{\mathcal{H}}.\end{aligned} \tag{7}$$

The proof is elementary, using the properties of the inner product and of the RKHS embedding.

Lemma 1 quantifies how well $\tilde{\mu}_{T+1}$ can serve as a drop-in replacement of μ_{T+1}. The introduced error will be small if all three terms on the right hand side are small. For the first term, we know that this is the case when the number of samples in S_T is large enough, since $\|\mathbf{A}\|_{\mathcal{F}}\|$ is a constant, and we know that the empirical distribution, $\hat{\mu}_T$, converges to the true distribution, μ_T. Similarly, the second terms becomes small in the limit of many samples set and many samples per set, because we know that the estimated operator, $\tilde{\mathbf{A}}$, converges to the operator of the true dynamics, $\mathbf{A}$, in this case. Consequently, EDD will provide a good estimate of the next distribution time step, given enough data and if our assumptions about the distribution evolution are fulfilled (i.e. $\|\epsilon_t\|$ is small).

d) Generating a Sample Set by Herding. Equation (6) suggests a way for associating a set of weighted samples,

$$\tilde{S}_{T+1} = \bigcup_{t=2}^{T} \Big\{ \frac{\beta_t}{n_t} \cdot z_1^t,\; \ldots,\; \frac{\beta_t}{n_t} \cdot z_{n_t}^t \Big\}, \tag{8}$$

with $\tilde{\mu}_{T+1}$, where $a \cdot b$ indicates not multiplication but that the sample b appears with a weight a. As we show in Section 5, this representation suffices for many purposes, in particular for learning a maximum-margin classifier. Other situations, however, might require a representation of $\tilde{\mu}_{T+1}$ by uniformly weighted samples, i.e. a set $\bar{S}_{T+1} = \{\bar{z}_1, \dots, \bar{z}_m\}$ such that $\tilde{\mu}_{T+1} \approx \mu(\bar{S}_{T+1}) = \frac{1}{m}\sum_{i=1}^{m} \phi(\bar{z}_i)$. To obtain such a set we use the RKHS variant of *herding* [4], a deterministic procedure for approximating a probability distribution by a set of samples. For any embedded distribution, $\eta \in \mu(\mathcal{P})$, herding constructs a sequence of samples, $\bar{z}_1, \bar{z}_2, \dots$, by the following rules,

$$\bar{z}_1 = \operatorname*{argmax}_{z \in \mathcal{Z}} \left\langle \phi(z), \eta \right\rangle_{\mathcal{H}}, \tag{9}$$

$$\bar{z}_n = \operatorname*{argmax}_{z \in \mathcal{Z}} \left\langle \phi(z), \eta - \frac{1}{n}\sum_{i=1}^{n-1} \phi(\bar{z}_i) \right\rangle_{\mathcal{H}}, \text{ for } n \geq 2.$$

Herding can be understood as an iterative greedy optimization procedure for finding examples $\bar{z}_1, \dots, \bar{z}_n$ that minimize $\|\eta - \frac{1}{n}\sum_{i=1}^{n} \phi(\bar{z}_i)\|_{\mathcal{H}}$ [2]. This interpretation shows that the target vector, η, is not restricted to be an embedded distribution, so herding can be applied to arbitrary vectors in $\mathcal{H}$. Doing so for $\tilde{\mu}_{T+1}$ yields a set $\bar{S}_{T+1} = \{\bar{z}_1, \dots, \bar{z}_{n_{T+1}}\}$ that can act as a drop-in replacement for an actual training set S_{T+1}. However, it depends on the concrete task whether it is possible to compute $\bar{S}_{T+1}$ in practice, because it requires solving multiple pre-image problems (9), which could be computationally intractable. A second interesting aspect of herding is that for any $\eta \in \mathcal{H}$, the herding approximation always has a pre-image in $\mathcal{P}$ (the empirical distribution defined by the herded samples). Therefore, herding can also be interpreted as an approximate projection from $\mathcal{H}$ to $\mu(\mathcal{P})$.

In Algorithm 1 we provide pseudo-code for EDD. It also shows that despite its mathematical derivation, the actual algorithms is easy to implement and execute.

2.2. Extension to Non-Uniform Weights

Our above description of EDD, in particular Equation (3), treats all given samples sets as equally important. In practice, this might not be desirable, and one might want to put more emphasis on some terms in the regression than on others. This effect can be achieved by introducing a weight, γ_t, for each of the summands of the least-squares problems (3). Typical choices are $\gamma_t = \rho^{-t}$, for a constant $0 < \rho < 1$, which expresses a belief that more recent observations are more trustworthy than earlier ones, or $\gamma_t = \sqrt{n_t}$, which encodes that the mean embedding of a sample set is more reliable if the set contains more samples.

As in ordinary least squares regression, per-term weights impact the coefficient matrix W, and thereby the concrete expressions for β_t. However, they do not change the overall structure of $\tilde{\mu}_{T+1}$ as a weighted combination of the observed data, so herding and PredSVM (see Section 5) training remain possible without structural modifications.

Algorithm 1 Extrapolating the distribution dynamics

input kernel function $k : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$
input sets $S_t = \{z_1^t, \dots, z_{n_t}^t\} \subset \mathcal{Z}$ for $t = 1, \dots, T$
input regularization parameter $\lambda \geq 0$

$K \leftarrow (T-1)\times(T-1)$-matrix with entries
$K_{st} = \frac{1}{n_s n_t}\sum_{i=1}^{n_s}\sum_{j=1}^{n_t} k(z_i^s, z_j^t)$
$\kappa \leftarrow (T-1)$-vector with entries
$\kappa_s = \frac{1}{n_s n_T}\sum_{i=1}^{n_s}\sum_{j=1}^{n_T} k(z_i^s, z_j^T)$
$\beta^* \leftarrow (K + \lambda I)^{-1}\kappa \quad \in \mathbb{R}^{T-1}$

output weighted sample set
$\tilde{S}_{T+1} = \bigcup_{t=2}^{T} \left\{ \frac{\beta_t}{n_t} \cdot z_1^t, \dots, \frac{\beta_t}{n_t} \cdot z_{n_t}^t \right\}$, with $\beta_t = \beta_{t-1}^*$

optional Herding step:
input output size m
$\bar{z}_1 \leftarrow \operatorname{argmax}_{z \in \mathcal{Z}} \sum_{t=2}^{T} \frac{\beta_t}{n_t}\sum_{i=1}^{n_t} k(z, z_i^t)$
for $n = 2, \dots, m$ **do**
$\bar{z}_n \leftarrow \operatorname*{argmax}_{z \in \mathcal{Z}} \left[\sum_{t=2}^{T} \frac{\beta_t}{n_t}\sum_{i=1}^{n_t} k(z, z_i^t) - \frac{1}{n}\sum_{i=1}^{n-1} k(z, \bar{z}_i) \right]$
end for
output sample set $\bar{S}_{T+1} = \{\bar{z}_1, \dots, \bar{z}_m\}$

3. Related Work

To our knowledge, the problem of extrapolating a time-varying probability distribution from a set of samples has not been studied in the literature before. However, a large of body work exists that studies related problems or uses related techniques.

The prediction of future states of a dynamical system or time-variant probability distribution is a classical application of probabilistic state space models, such as *Kalman filters* [10], and *particle filters* [7]. These techniques aim at modeling the probability of a time-dependent system jointly over all time steps. This requires observed data in the form of time series, e.g. trajectories of moving particles [24]. EDD, on the other hand, learns only the transitions between the marginal distribution at one point of time to the marginal distribution at the next point of time. For this, independent sample sets from different time points are sufficient. The difference between both approaches become apparent, e.g., by looking at a system of homogeneously distributed particles that rotate around a center. A joint model would learn the circular orbits, while EDD would learn the identity map, since the data distributions are the same at any time.

A related line of existing work aims at predicting the future motion of specific objects in videos [11, 25, 28, 29], or anticipating the behavior of a human in order to facilitate interaction with a robot [12, 26]. Such model-based approaches rely on task-specific assumptions, e.g. on the visual appearance of objects. This allows them to make pre-

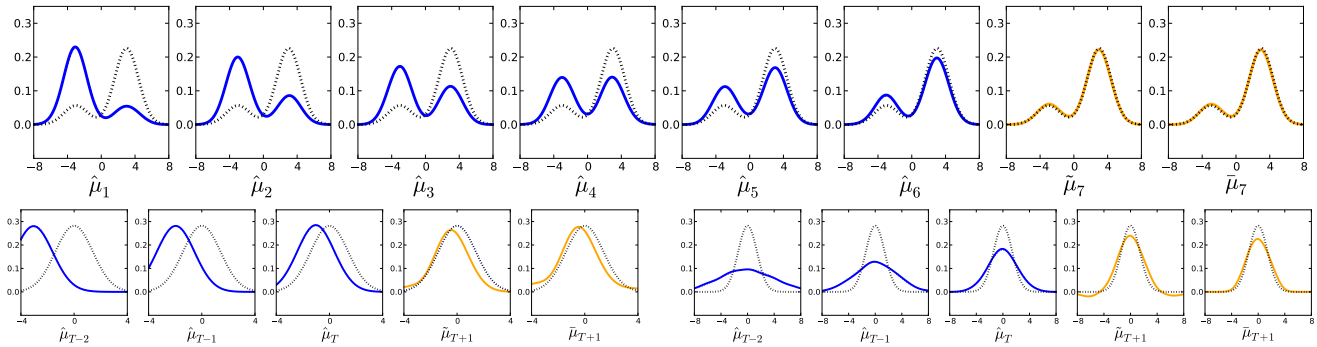

Figure 2. Illustration of Experiment 1: mixture of Gaussians with changing proportions (top), translating Gaussian (bottom left) and Gaussian with contracting variance (bottom right). Blue curves illustrate the RKHS embeddings $\hat{\mu}_1, \ldots, \hat{\mu}_T$ of the given sample sets ($n = 1000$). The orange curves are EDD's prediction of the distribution at time $T+1$, as output of the learned operator ($\tilde{\mu}_T{+}1$) and after additional herding ($\bar{\mu}_T{+}1$).

cise predictions about possible model states at future times, but renders them not applicable to our situation of interest, where the goal is to predict the future behavior of a probability distribution, not of an individual object.

In the literature of *RKHS embeddings*, a line of work related to EDD is the learning of *conditional distributions* by means of covariance operators, which has also be interpreted as a vector-valued regression task [14]. Given a current distribution and such a conditional model, one could infer the marginal distribution of the next time step [21]. Again, the difference to EDD lies in the nature of the modeled distribution and the training data required for this. To learn conditional distributions, the training data must consist of pairs of data points at two subsequent time points (essentially a minimal trajectory), while in the scenario we consider correspondences between samples at different time points are not available and often would not even make sense. For example, in Section 4 we apply EDD to images of car models from different decades. Correspondence between the actual cars depicted in such images do not exist.

4. Experiments

We report on experiments on synthetic and real data in order to highlight the working methodology of EDD, and to show that extrapolating the distribution dynamics is possible for real data and useful for practical tasks.

Experiment 1: Synthetic Data. First, we perform experiments on synthetic data for which we know the true data distribution and dynamics, in order to highlight the working methodology of EDD. In each case, we use sample sets of size $n = \{10, 100, 1000\}$ and we use a regularization constant of $\lambda = \frac{1}{n}$. Where possible we additionally analytically look at the limit case $n \to \infty$, i.e. $\hat{\mu}_t = \mu_t$. For the RKHS embedding we use a Gaussian kernel with unit variance.

First, we set $d_t = \alpha_t \mathcal{N}(3; 1) + (1-\alpha)\mathcal{N}(-3; 1)$, a mixture of Gaussians distribution with mixture coefficients that vary over time as $\alpha_t \in \{0.2, 0.3, \ldots, 0.8\}$. Figure 2 (top) illustrates the results: trained on the first six samples sets (blue lines), the prediction by EDD (orange) match almost perfectly the seventh (dashed), with or without herding. In order to interpret this result, we first observe that due the form of the distributions d_t it is not surprising that μ_{T+1} could be expressed as linear combination of the $\mu_1, \ldots, \mu_T$, provided we allow for negative coefficients. What the result shows, however, is that EDD is indeed able to find the right coefficients from the sample sets, indicating that the use of an autoregressive model is justified in this case.

The prediction task can be expected to be harder if not only the values of the density change between time steps but also the support. We test this by setting $d_t = \mathcal{N}(T + 1 - t; 1)$, i.e. a Gaussian with shifting location of the mean, which we call the *translation* setting. Figure 2 (bottom left) illustrates the last three steps of the total nine observed steps of the dynamics (blue) and its extrapolation (orange) with and without herding. One can see that EDD indeed is able to extrapolate the distribution to a new region of the input space: the modes of the predicted $\tilde{\mu}_{T+1}$ and $\bar{\mu}_{T+1}$ lie right of the mode of all inputs. However, the prediction quality is not as good as in the mixture setting, indicating that this is in fact a harder task.

Finally, we study a situation where it is not clear on first sight whether the underlying dynamics has a linear model: a sequence of Gaussians with decreasing variances, $d_t = \mathcal{N}(0; T - t + 1)$, which we call the *concentration* setting. The last three steps of the nine observed steps of the dynamics and its extrapolation with and without herding are illustrated in Figure 2 (bottom right) in blue and orange, respectively. One can see that despite the likely nonlinearity, EDD is able to predict a distribution that is more concentrated (has lower variance) than any of the inputs. In this case, we also observe that the predicted density function exhibits negative values, and that Herding removes those.

As a quantitative evaluation we report in Tables 1 and 2 how well the predicted distributions correspond to

(a) *Mixture* setting

n	*true dist.*	last obs.	EDD	EDD+H
10	0.13 ± 0.03	0.13 ± 0.03	0.17 ± 0.02	0.18 ± 0.03
100	0.03 ± 0.01	0.07 ± 0.01	0.05 ± 0.02	0.05 ± 0.02
1000	< 0.01	0.06 ± 0.00	≤ 0.01	< 0.01
∞	0.00	0.07	0.00	—

(b) *Translation* setting

n	*true dist.*	last obs.	EDD	EDD+H
10	0.13 ± 0.12	0.28 ± 0.21	0.31 ± 0.17	0.27 ± 0.18
X 100	0.04 ± 0.04	0.27 ± 0.12	0.20 ± 0.10	0.18 ± 0.11
1000	0.01 ± 0.01	0.26 ± 0.07	0.14 ± 0.06	0.13 ± 0.06
∞	0.00	0.27	0.09	—

(c) *Concentration* setting

n	*true dist.*	last obs.	EDD	EDD+H
10	0.13 ± 0.12	0.25 ± 0.20	0.32 ± 0.17	0.32 ± 0.18
100	0.04 ± 0.04	0.20 ± 0.10	0.22 ± 0.12	0.22 ± 0.12
1000	0.01 ± 0.01	0.19 ± 0.06	0.15 ± 0.07	0.15 ± 0.07
∞	0.00	0.19	0.07	—

Table 1. Approximation quality of EDD, EDD with herding (EDD +H) and baselines measured in RKHS norm (lower values are better) for different synthetic settings. For details, see Section 4.

the ground truth ones as measured by the Hilbert space (HS) distance and the Kullback-Leibler divergences, respectively. The latter is only possible for EDD after herding, when the prediction is a proper probability distribution (non-negative and normalized). Besides EDD, we include the baseline of reusing the last observed sample set as a proxy for the next one. To quantify how much of the observed distance is due to the prediction step and how much is due to an unavoidable sampling error, we also report the values for a sample set S_{T+1} of the same size from the true distribution d_{T+1}.

The results confirm that, given sufficiently many samples of the earlier tasks, EDD is indeed able to successfully predict the dynamics of the distribution. The predicted distribution $\tilde{\mu}_{T+1}$ is closer to the true distribution μ_{T+1} than the most similar observed distribution, $\hat{\mu}_T$. For *translation* and *concentration*, the analytic results show that even for $n \to \infty$ the difference is non-zero, suggesting that the true dynamics are not exactly linear in the RKHS. However, the residual is small compared to the measured quantities.

Experiment 2: Real World Data. In a second set of experiments, we test EDD's suitability for real data by applying to video sequences from [6]. The dataset consists of 1121 video sequences of six semantic categories, *birthday, parade, picnic, show, sports*, and *wedding*, from two sources, *Kodak* and *YouTube*. Each video is represented by a collection of spatio-temporal interest points (STIPs) with

(a) *Mixture* setting

n	*true dist.*	last obs.	EDD+H
10	0.08 ± 0.05	0.08 ± 0.03	0.17 ± 0.05
100	< 0.01	0.03 ± 0.01	0.02 ± 0.01
1000	< 0.001	0.02 ± 0.00	< 0.005

(b) *Translation* setting

n	*true dist.*	last obs.	EDD+H
10	0.05 ± 0.05	0.29 ± 0.19	0.28 ± 0.10
100	< 0.005	0.26 ± 0.05	0.11 ± 0.04
1000	< 0.001	0.25 ± 0.02	0.07 ± 0.02

(c) *Concentration* setting

n	*true dist.*	last obs.	EDD+H
10	0.05 ± 0.05	0.23 ± 0.13	0.56 ± 0.18
100	< 0.005	0.16 ± 0.04	0.20 ± 0.06
1000	< 0.001	0.16 ± 0.01	0.06 ± 0.02

Table 2. Approximation quality of EDD, EDD with herding (EDD +H) and baselines measured by KL divergence (lower values are better) for different synthetic settings. For details, see Section 4.

162-dimensional feature vectors.[1] For each video, except six that are less than one second long, we split the STIPs into groups by creating segments of 10 frames each. Different segments have different numbers of samples, because the STIPs are obtained from the response of an interest operator. Different video also show a strong diversity in this characteristics: the number of STIPs per segment varies between 1 and 550, and the number of segments per video varies between 3 and 837.

As experimental setup, we use all segments of a movie except the last one as input sets for EDD, and we measure the distance between the predicted next distribution and the actual last segment. Table 3 shows the results split by data source and category for two choices of kernels: the *RBF-χ^2 kernel*, $k(z, \bar{z}) = \exp(-\frac{1}{2}\chi^2(z, \bar{z}))$ for $\chi^2(z, \bar{z}) = \frac{1}{d}\sum_{i=1}^{d} \frac{(z_i - \bar{z}_i)^2}{\frac{1}{2}(z_i + \bar{z}_i)}$, and the *histogram intersection kernel*, $k(z, \bar{z}) = \frac{1}{d}\sum_{i=1}^{d} \min(z_i, \bar{z}_i)$, both for $z, \bar{z} \in \mathbb{R}_+^d$. For each data source and category we report the average and standard error of the Hilbert-space distance between distribution. As baselines, we compare against re-using the last observed segment, i.e. not extrapolating, and against the distribution obtained from merging all segments, i.e. the global video distribution. One can see that the predictions by EDD are closer to the true evolution of the videos than both baselines in all cases but two, in which it is tied with using the last observation. The improvement is statistically significant (bold print) to a 0.05 level according to Wilcoxon signed rank test with multi-test correction, except for some cases with only few sequences.

[1] http://vc.sce.ntu.edu.sg/index_files/VisualEventRecognition/features.html

(a) Histogram intersection kernel

YouTube	EDD	last seg.	all seg.
birthday(151)	$\mathbf{0.15 \pm 0.005}$	0.16 ± 0.006	0.16 ± 0.005
parade(119)	$\mathbf{0.13 \pm 0.006}$	0.15 ± 0.008	0.15 ± 0.007
picnic(85)	$\mathbf{0.13 \pm 0.007}$	0.15 ± 0.009	0.15 ± 0.008
show(200)	$\mathbf{0.14 \pm 0.004}$	0.15 ± 0.005	0.16 ± 0.005
sports(258)	$\mathbf{0.14 \pm 0.004}$	0.15 ± 0.004	0.16 ± 0.004
wedding(90)	$\mathbf{0.16 \pm 0.007}$	0.17 ± 0.009	0.18 ± 0.007
Kodak	EDD	last seg.	all seg.
birthday(16)	$\mathbf{0.17 \pm 0.015}$	0.20 ± 0.022	0.18 ± 0.014
parade(14)	0.15 ± 0.023	0.17 ± 0.030	0.17 ± 0.022
picnic(6)	0.17 ± 0.028	0.17 ± 0.031	0.20 ± 0.027
show(55)	$\mathbf{0.20 \pm 0.011}$	0.23 ± 0.013	0.21 ± 0.011
sports(74)	0.15 ± 0.006	0.16 ± 0.007	0.16 ± 0.007
wedding(27)	$\mathbf{0.18 \pm 0.011}$	0.21 ± 0.013	0.19 ± 0.011

(b) RBF-χ^2 kernel

YouTube	EDD	last seg.	all seg.
birthday(151)	$\mathbf{0.17 \pm 0.006}$	0.19 ± 0.007	0.19 ± 0.006
parade(119)	$\mathbf{0.15 \pm 0.007}$	0.17 ± 0.009	0.18 ± 0.008
picnic(85)	$\mathbf{0.15 \pm 0.008}$	0.17 ± 0.010	0.18 ± 0.010
show(200)	$\mathbf{0.17 \pm 0.005}$	0.18 ± 0.005	0.20 ± 0.006
sports(258)	$\mathbf{0.16 \pm 0.004}$	0.17 ± 0.005	0.20 ± 0.005
wedding(90)	$\mathbf{0.18 \pm 0.008}$	0.20 ± 0.010	0.21 ± 0.008
Kodak	EDD	last seg.	all seg.
birthday(16)	0.20 ± 0.018	0.24 ± 0.027	0.22 ± 0.015
parade(14)	0.18 ± 0.027	0.20 ± 0.035	0.20 ± 0.025
picnic(6)	0.19 ± 0.029	0.19 ± 0.032	0.24 ± 0.031
show(55)	$\mathbf{0.23 \pm 0.012}$	0.26 ± 0.015	0.25 ± 0.013
sports(74)	$\mathbf{0.17 \pm 0.007}$	0.18 ± 0.008	0.20 ± 0.008
wedding(27)	0.22 ± 0.012	0.24 ± 0.015	0.23 ± 0.013

Table 3. Experiment 2: Distance between last video segment and its prediction by EDD, the last observed segment (last seg.) and the union of all segments (all seg.). Values in parentheses after the class names specify the number of test sequences.

5. Application: Predictive Domain Adaptation

We are convinced that being able to extrapolate a time-varying probability distribution into the future will be useful for numerous practical applications. As an illustrative example, we look at one specific problem: learning a classifier under distribution drift, when for training the classifier data from the time steps $t = 1, \dots, T$ is available, but by the time the classifier is applied to its target data, the distribution has moved on to time $t = T+1$. A natural choice to tackle this situation would be the use of *domain adaptation* techniques [9]. However, those typically require that at least unlabeled data from the target distribution is available, which in practice might not be the case. For example, in an online prediction setting, such as spam filtering, predictions need to be made on the fly. One cannot simply stop, collect data from the new data distribution, and retrain the classifiers. Instead, we show how EDD can be used to train a maximum margin classifier for data distributed according to d_{T+1}, with only data from d_1 to d_T available. We call this setup *predictive domain adaptation (PDA)*.

To our knowledge, the PDA problem has not appeared in the literature before. Recent work on continuous domain adaptation [8, 27] has studied the problem of classification under evolving distributions, but in a setting opposite to ours: a static source distribution but a target distribution that changes over time. We plan to study in future work if both viewpoints can be unified.

Let $S_t = \{(x_1^t, y_1^t), \dots, (x_{n_t}^t, y_{n_t}^t)\}$ for $t = 1, \dots, T$, be a sequence of labeled training sets, where $\mathcal{X}$ is the input space, e.g. images, and $\mathcal{Y} = \{1, \dots, K\}$ is the set of class labels. For any kernel $k_{\mathcal{X}}(x, \bar{x})$ on $\mathcal{X}$, we form a joint kernel, $k((x, y), (\bar{x}, \bar{y})) = k_{\mathcal{X}}(x, \bar{x})[\![y = \bar{y}]\!]$ on $\mathcal{X} \times \mathcal{Y}$ and we apply EDD for $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$. The result is an estimate of the next time step of the joint probability distribution, $d_{T+1}(x, y)$, as a vector, $\tilde{\mu}_{T+1}$, or in form of a weighted sample set, $\tilde{S}_{T+1}$.

To see how this allows us to learn a better adapted classifier, we first look at the situation of binary classification with 0/1-loss function, $\ell(y, \bar{y}) = [\![y \neq \bar{y}]\!]$. If a correctly distributed training S_{T+1} of size n_{T+1} were available, one would aim for minimizing the regularized risk functional

$$\frac{1}{2}\|w\|^2 + \frac{C}{n_{T+1}} \sum_{(x_i, y_i) \in S_{T+1}} \ell(\, y_i, \operatorname{sign}\langle w, \psi(x_i)\rangle\,), \qquad (10)$$

where C is a regularization parameter and ψ is any feature map, not necessarily the one induced by $k_{\mathcal{X}}$. To do so numerically, one would bound the loss by a convex surrogate, such as the hinge loss, $\max\{0, 1 - y\langle w, \psi(x)\rangle\}$, which make the overall optimization problem convex and therefore efficiently solvable to global optimality.

In the PDA situation, we do not have a training set S_{T+1}, but we do have a prediction $\tilde{S}_{T+1}$ provided by EDD in the form of Equation (8). Therefore, instead of the empirical average in (10), we can form a predicted empirical average using the weighted samples in $\tilde{S}_{T+1}$. This leads to the *predicted regularized risk functional*,

$$\frac{1}{2}\|w\|^2 + C \sum_{t=2}^{T} \frac{\beta_t}{n_t} \sum_{i=1}^{n_t} \ell(\, y_i^t, \operatorname{sign}\langle w, \psi(x_i^t)\rangle\,), \qquad (11)$$

that we would like to minimize. In contrast to the expression (10), replacing the 0/1-loss by the hinge loss does not lead to a convex upper bound of (11), because the coefficients β_t can be either positive or negative. However, we can use that $\ell(\, y, \bar{y}\,) = 1 - \ell(\, -y, \bar{y}\,)$, and obtain an equivalent expression for the loss term with only positive weights, $\sum_{t=2}^{T} b_t \sum_{i=1}^{n_t} \ell(\, \bar{y}_{ti}, \operatorname{sign} f(x_{ti})\,) + c$, where $b_t = |\beta_t|/n_t$ and $\bar{y}_{ti} = (\operatorname{sign} \beta_t) y_{ti}$. The constant $c = \sum_t \beta_t$ plays no role for the optimization procedure, so we drop it from the

Figure 3. Example images from *CarEvolution* dataset [18]. The goal is to classify images by their manufacturer (BMW, Mercedes, VW). Each block shows one image from each the four groups: *1970s* (top left), *1980s* (top right), *1990s* (bottom left), and *later* (bottom right).

notation for the rest of this section. Now bounding each 0/1-loss term by the corresponding Hinge loss yields a convex upper bound of the predicted risk,

$$\frac{1}{2}\|w\|^2 + C\sum_{t=2}^{T} b_t \sum_{i=1}^{n_t} \max\{0, 1-\bar{y}_{ti}\langle w, \psi(x_i^t)\rangle\}. \quad (12)$$

Minimizing it corresponds to training a support vector machine with respect to the predicted data distribution, which we refer to as PredSVM. It can be done by any SVM package that support per-sample weights, e.g. libSVM [3]. A multi-class classifier can be obtained from this by the usual one-vs-rest or one-vs-one constructions.

Experiment 3: Predictive Domain Adaptation. To demonstrate the usefulness of training a classifier on a predicted data distribution, we perform experiments on the *CarEvolution* [18] data set.[2] It consists of 1086 images of cars, each annotated by the car manufacturer (*BMW*, *Mercedes* or *VW*) and the year in which the car model was introduced (between 1972 and 2013).

The data comes split into source data (years 1972–1999) and target data (years 2000-2013). We split the *source* part further into three decades: 1970s, 1980s, 1990s. Given these groups, our goal is to learn a linear PredSVM to distinguish between the manufacturers in the *target* part. For a second set of experiments we split the target set further into models from the 2000s and models from the 2010s, and we learn a linear PredSVM with the 1970s as target and the other tasks in inverse order as sources. As baseline, we use SVMs that were trained on any of the observed tasks, as well as an SVMs trained all the union of all source tasks. In all cases we choose the SVM parameter, $C \in \{10^0, \ldots, 10^6\}$, by five-fold cross validation on the respective training sets.

Table 4 summarizes the results for two different feature representations: Fisher vectors [17] and L^2-normalized DeCAF(fc6) features [5]. In all cases PredSVM improves the prediction quality over the other baselines, except once where training on all observed data is more effective.

[2] http://homes.esat.kuleuven.be/~krematas/VisDA/CarEvolution.html

method	FVs	decaf
1970s $\rightarrow \geq$2000s	39.1%	38.2%
1980s $\rightarrow \geq$2000s	43.8%	44.9%
1990s $\rightarrow \geq$2000s	49.0%	51.2%
all $\rightarrow \geq$2000s	51.2%	52.1%
PredSVM (temporal order)	51.5%	56.2%
method	FVs	decaf
2010s $\rightarrow$ 1970s	33.5%	34.0%
2000s $\rightarrow$ 1970s	31.6%	42.7%
1990s $\rightarrow$ 1970s	45.6%	50.0%
1980s $\rightarrow$ 1970s	44.7%	29.1%
all $\rightarrow$ 1970s	49.0%	49.0%
PredSVM (reverse order)	48.1%	50.5%

Table 4. Classification accuracy of PDA-SVM and baseline methods on *CarEvolution* data set (higher is better). Top: temporal order, bottom: reverse order. See Section 4 for details.

6. Summary and Discussion

In this work, we have introduced the task of predicting the future evolution of a time-varying probability distribution. We described a method that, given a sequence of observed samples set, extrapolates the distribution dynamics by one step. Its main components are two recent techniques from machine learning: the embeddings of probability distributions into a Hilbert space, and vector-valued regression. We also showed how the predicted distribution can be used to learn a classifier for a data distribution from which no training examples are available, not even unlabeled ones.

Our experiments on synthetic and real data gave insight into the working methodology of EDD and showed that it can –to some extend– predict the next state of a time-varying distribution from samples of earlier time steps, and that this can be useful for learning better classifiers. One shortcoming of our current method is its restriction to equally spaced time steps and that it extrapolates only by a single time unit. We plan to extend our framework to more flexible situations, e.g. continuous-time dynamics.

Acknowledgements

This work was funded in parts by the European Research Council under the European Unions Seventh Framework Programme (FP7/2007-2013)/ERC grant agreement no 308036.

References

[1] Y. Altun and A. Smola. Unifying divergence minimization and statistical inference via convex duality. In *Workshop on Computational Learning Theory (COLT)*, 2006. 2

[2] F. Bach, S. Lacoste-Julien, and G. Obozinski. On the equivalence between herding and conditional gradient algorithms. In *International Conference on Machine Learing (ICML)*, 2012. 4

[3] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. 8

[4] Y. Chen, M. Welling, and A. J. Smola. Super-samples from kernel herding. In *Uncertainty in Artificial Intelligence (UAI)*, 2010. 4

[5] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. DeCAF: A deep convolutional activation feature for generic visual recognition. In *International Conference on Machine Learing (ICML)*, 2014. 8

[6] L. Duan, D. Xu, I.-H. Tsang, and J. Luo. Visual event recognition in videos by learning from web data. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(9):1667–1680, 2012. 6

[7] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE Proceedings F (Radar and Signal Processing)*, 1993. 4

[8] J. Hoffman, T. Darrell, and K. Saenko. Continuous manifold based adaptation for evolving visual domains. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 7

[9] J. Jiang. A literature survey on domain adaptation of statistical classifiers. `http://sifaka.cs.uiuc.edu/jiang4/domain_adaptation/survey`, 2008. 7

[10] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82:35–45, 1960. 4

[11] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, and M. Hebert. Activity forecasting. In *European Conference on Computer Vision (ECCV)*, pages 201–214, 2012. 4

[12] H. S. Koppula and A. Saxena. Anticipating human activities using object affordances for reactive robotic response. In *Robotics: Science and Systems*, 2013. 4

[13] C. H. Lampert. Kernel methods in computer vision. *Foundations and Trends in Computer Graphics and Vision*, 4(3):193–285, 2009. 2

[14] G. Lever, L. Baldassarre, S. Patterson, A. Gretton, M. Pontil, and S. Grünewälder. Conditional mean embeddings as regressors. In *International Conference on Machine Learing (ICML)*, 2012. 5

[15] H. Lian. Nonlinear functional models for functional responses in reproducing kernel Hilbert spaces. *Canadian Journal of Statistics*, 35(4):597–606, 2007. 2, 3

[16] C. A. Micchelli and M. Pontil. On learning vector-valued functions. *Neural Computation*, 17(1):177–204, 2005. 2

[17] F. Perronnin, J. Sánchez, and T. Mensink. Improving the Fisher kernel for large-scale image classification. In *European Conference on Computer Vision (ECCV)*, 2010. 8

[18] K. Rematas, B. Fernando, T. Tommasi, and T. Tuytelaars. Does evolution cause a domain shift? In *ICCV Workshop on Visual Domain Adaptation and Dataset Bias*, 2013. 8

[19] A. Smola, A. Gretton, L. Song, and B. Schölkopf. A Hilbert space embedding for distributions. In *International Conference on Algorithmic Learning Theory (ALT)*, 2007. 2

[20] L. Song. *Learning via Hilbert space embedding of distributions*. PhD thesis, University of Sydney, 2008. 2

[21] L. Song, J. Huang, A. Smola, and K. Fukumizu. Hilbert space embeddings of conditional distributions with applications to dynamical systems. In *International Conference on Machine Learing (ICML)*, 2009. 5

[22] B. K. Sriperumbudur, K. Fukumizu, and G. Lanckriet. Universality, characteristic kernels and RKHS embedding of measures. *Journal of Machine Learning Research (JMLR)*, 12:2389–2410, 2011. 1

[23] Z. Szabo, A. Gretton, B. Poczos, and B. Sriperumbudur. Learning theory for distribution regression. arXiv:1411.2066 [math.ST], 2014. 3

[24] R. Talmon and R. R. Coifman. Empirical intrinsic geometry for nonlinear modeling and time series filtering. *Proceedings of the National Academy of Sciences (PNAS)*, 110(31):12535–12540, 2013. 4

[25] J. Walker, A. Gupta, and M. Hebert. Patch to the future: Unsupervised visual prediction. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 4

[26] Z. Wang, C. H. Lampert, K. Mülling, B. Schölkopf, and J. Peters. Learning anticipation policies for robot table tennis. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011. 4

[27] J. Xu, S. Ramos, D. Vázquez, and A. M. López. Incremental domain adaptation of deformable part-based models. In *British Machine Vision Conference (BMVC)*, 2014. 7

[28] K. Yamaguchi, A. C. Berg, L. E. Ortiz, and T. L. Berg. Who are you with and where are you going? In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. 4

[29] B. D. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. Bagnell, M. Hebert, A. K. Dey, and S. Srinivasa. Planning-based prediction for pedestrians. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2009. 4