Universum Learning for SVM Regression

Sauptik Dhar
Research and Technology Center
Robert Bosch LLC
Palo Alto, CA, 94304, USA
sauptik.dhar@us.bosch.com

Vladimir Cherkassky
Department of Electrical and Computer Engineering
University of Minnesota
Minneapolis, MN 55455, USA
cherk001@umn.edu

***Abstract*— This paper extends the idea of Universum learning to regression problems. We propose new Universum-SVM formulation for regression problems that incorporates a priori knowledge in the form of additional data samples. These additional data samples, or Universum samples, belong to the same application domain as the training samples, but they follow a different distribution. Several empirical comparisons are presented to illustrate the utility of the proposed approach.**

Keywords*— *Support Vector Regression; learning through contradiction; Universum Learning

I. Introduction

The technique of Universum learning or *learning through contradiction* [1, 2] provides a formal mechanism for incorporating a priori knowledge about the application domain, in the form of additional (unlabeled) Universum samples. Universum learning has been originally introduced for binary classification problems [2, 3] and it has been shown to be particularly effective for high-dimensional low-sample size data settings [3, 4, 5]. More recently, Universum learning has been extended to various non-standard classification settings [6-15]. However, most research on Universum learning has been limited to binary classification problems. It is not clear how to extend the idea of Universum learning to other types of learning problems.

This paper describes Universum learning for supervised learning problem known as *regression* or real-valued function estimation from noisy samples [1, 2, 15 - 18]. Under regression setting, a real-valued output can be represented as the sum of (unknown) target function $t(\mathbf{x})$ and an additive error term δ:

$$y = t(\mathbf{x}) + \delta \tag{1}$$

Here, data samples $(\mathbf{x}, y)$ follow an underlying (unknown) distribution described by the joint density function,

$$p(\mathbf{x}, y) = p(y \mid \mathbf{x}) p(\mathbf{x}) \tag{2}$$

So the target function is the mean of the output conditional probability,

$$t(\mathbf{x}) = \int y p(y \mid \mathbf{x})\, d\mathbf{x} \tag{3}$$

The goal of learning is to estimate the '*best*' function (model) from a set of approximating functions $f(\mathbf{x}, \omega)$ parameterized by $\omega \in \Omega$. The quality of the approximation is measured by a given loss function $L(y, f(\mathbf{x}, \omega))$. Thus, the problem of regression involves estimation of a real-valued function that minimizes the risk functional,

$$R(\omega) = \int L(y, f(\mathbf{x}, \omega)) p(\mathbf{x}, y) d\mathbf{x} dy \tag{4}$$

A common loss function for regression problems is the squared loss,

$$L(y, f(\mathbf{x}, \omega)) = (y - f(\mathbf{x}, \omega))^2 \tag{5}$$

The difficulty of regression estimation is due to the fact that statistical distribution of $(\mathbf{x}, y)$ is unknown, and the only information (about it) is available in the form of finite training data set $(\mathbf{x}_i, y_i)_{i=1}^n$. For regression problems, one can also expect to achieve improved generalization performance by incorporating a priori knowledge in the form of additional Universum samples. This paper introduces the concept of Universum learning for regression problems and provides new optimization formulation that incorporates additional Universum data into standard SVM regression setting.

This paper is organized as follows. Section II reviews standard SVM regression (SVR) formulation [2, 18]. Section III introduces the new Universum-SVM regression (U-SVR) formulation. Section IV describes a new technique for understanding the effectiveness of U-SVR learning. Next we provide empirical results to illustrate the effectiveness of this new formulation in Section V. Finally, Section VI presents conclusions.

II. SVM Regression

This section provides a brief description of standard SVM regression (SVR) formulation following Vapnik [1, 2]. This SVR setting uses special margin-based loss function known as ε - insensitive loss $L_\varepsilon(f(\mathbf{x}, \mathbf{w}), y) = \max(|y - f(\mathbf{x}, \mathbf{w})| - \varepsilon, 0)$. This loss function is shown in Fig. 1. Then SVR formulation can be stated as:

Given i.i.d training samples $(\mathbf{x}_i, y_i)_{i=1}^n$, with $\mathbf{x} \in \mathbb{R}^d$, $y \in \mathbb{R}$; the linear SVR model can be found by solving the optimization problem:

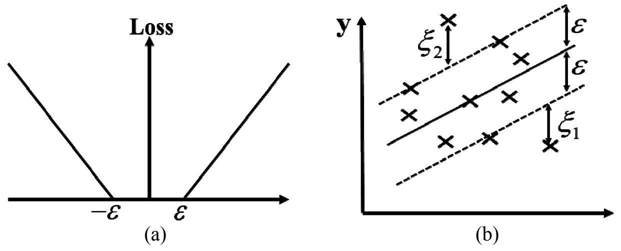


Fig. 1. SVM regression (a) ε - insensitive loss function. (b) slack variables ξ for linear SVM regression formulation.

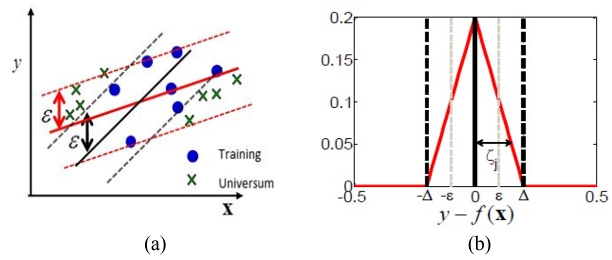


Fig. 2. U-SVM regression. (a) Two SVM regression models explain training data equally well, but have different number of contradictions on the Universum. The model with a larger number of contradictions (in black) is selected. (b) Loss function for the universum samples $U_\Delta(y_j^* - f(\mathbf{x}_j^*))$ (with Δ = 0.2 for illustration).

$$\min_{\mathbf{w},b,\xi,\xi^*} \quad \frac{1}{2}(\mathbf{w}\cdot\mathbf{w}) \;+\; C\sum_{i=1}^{n}(\xi_i+\xi_i^*) \tag{6}$$

$$\begin{aligned} s.t. \quad & y_i-(\mathbf{w}\cdot\mathbf{x}_i)-b\le\varepsilon+\xi_i \quad & \xi_i\ge 0 \quad & i=1\ldots n \\ & (\mathbf{w}\cdot\mathbf{x}_i)+b-y_i\le\varepsilon+\xi_i^* \quad & \xi_i^*\ge 0 \quad & i=1\ldots n \end{aligned}$$

Here n := number of training samples, and d := dimensionality of the input space (or the number of input variables). Note that training samples falling inside the ε - tube have zero loss, and samples outside the ε - insensitive zone are linearly penalized using the slack variables $\xi_i,\xi_i^*\ge 0$, $i=1\ldots n$ (as shown in Fig. 1b). These slack variables contribute to the empirical risk for the SVR formulation $R_{emp}(\mathbf{w})=\sum_{i=1}^{n}(\xi_i+\xi_i^*)$. The linear SVR formulation (6) has two tunable parameters that control model complexity, $C\ge 0$ and $\varepsilon\ge 0$. Parameter $C\ge 0$ controls the trade-off between the empirical risk and the penalization term. Parameter ε controls the size of the ε - insensitive zone in margin-based loss. Notably, using tunable (data-dependent) parameter ε effectively implements margin-based loss for regression setting, and thus enables complexity control independent of dimensionality. According to [18], SVR regression model can 'explain' all training data samples falling inside ε - insensitive zone, but cannot explain samples outside this zone. So the goal of learning is to model (explain) most of the training data using SVR loss function with 'small' ε. That is, using small ε -values (for regression) is similar to large margin (for classification) [18].

III. Universum-SVM Regression

A. *Universum-SVM Regression formulation*

Consider regression setting where available training data $(\mathbf{x}_i, y_i)_{i=1}^n$ is modeled using linear SVR. As described in Section II, for SVR the concept of 'margin' is implemented via ε - insensitive loss. That is, training samples falling inside ε - insensitive zone are 'explained' by SVR model, and samples falling outside 'falsify' or 'contradict' this model.

Next, consider two SVR models which explain training samples *equally well*, e.g. both SVR models use the same value of ε and achieve the same empirical risk $R_{emp}(\mathbf{w})=\sum_{i=1}^{n}(\xi_i+\xi_i^*)$ for training samples. For example, Fig. 2a shows two SVR models that explain available training data equally well, i.e. have zero empirical risk. Next, consider additional *Universum* samples $(\mathbf{x}_j^*, y_j^*)_{j=1}^m$. These samples are defined in the same $(\mathbf{x}, y)$ space as the training samples, and reflect a priori knowledge that they *should not* be explained well by SVR model, i.e. should *lie outside* the ε - insensitive tube. For the toy example shown in Fig. 2a, we should favor the model shown in black, for which most Universum samples cannot be explained by SVR model. Note that for regression setting, Universum samples are *labeled*, unlike unlabeled Universum samples for classification. This motivates the new Universum support vector regression (U-SVR) formulation where:

– Standard ε - insensitive loss is used for training samples.

– Universum samples are penalized by a different loss function as shown in Fig. 2b.

This new loss function forces the universum samples to lie 'far away' from the regression model, so that samples outside a $\pm\Delta$ zone have zero loss. Penalization of universum samples inside the $\pm\Delta$ zone is achieved via the slack variables ζ_j as shown in Fig. 2b. Note that the tunable parameter Δ can be larger (or smaller) than ε . This leads to the following optimization formulation for U-SVR:

Given i.i.d. training samples $(\mathbf{x}_i, y_i)_{i=1}^n$, with $\mathbf{x}\in\mathbb{R}^d$, $y\in\mathbb{R}$, and additional universum samples $(\mathbf{x}_j^*, y_j^*)_{j=1}^m$; the linear U-SVR model can be found by solving the optimization problem:

$$\min_{\mathbf{w},b,\xi,\xi^*,\zeta} \quad L(\mathbf{w},b,\xi,\xi^*,\zeta) = \tag{7}$$

(Training samples) **(Universum samples)**

$$\boxed{\begin{aligned} & \frac{1}{2}(\mathbf{w}\cdot\mathbf{w}) \;+\; C\sum_{i=1}^{n}(\xi_i+\xi_i^*) \\ s.t. \quad & y_i-(\mathbf{w}\cdot\mathbf{x}_i)-b\le\varepsilon+\xi_i \\ & (\mathbf{w}\cdot\mathbf{x}_i)+b-y_i\le\varepsilon+\xi_i^* \\ & \xi_i,\xi_i^*\ge 0, \quad i=1\ldots n \\ & \text{(convex)} \end{aligned}} \;+\; \boxed{\begin{aligned} & C^*\sum_{j=1}^{m}\zeta_j \\ & \left|y_j^*-(\mathbf{w}\cdot\mathbf{x}_j^*)-b\right|\le\Delta+\zeta_j \\ & \zeta_j\ge 0, \quad i=1\ldots n \\ & \text{(concave } U_\Delta(y_j-f(\mathbf{x}_j^*))\text{)} \end{aligned}}$$

Here, parameters: $\varepsilon, \Delta \ge 0$ and $C, C^* \ge 0$ control the tradeoff between 'explanation' of training samples and 'falsification' of the universum samples. This new optimization formulation for U-SVR has two additional tuning parameters: C^* and Δ, relative to standard SVR setting. Setting $C^* = 0$ or $\Delta = 0$ in U-SVR formulation yields standard SVR formulation (6).

B. Computational Implementation of U-SVR

The U-SVR formulation (7) is non-convex due to non-convexity of the Universum loss $U_\Delta(y_j - f(\mathbf{x}_j^*))$ shown in Fig. 2b. Hence, it cannot be solved using standard convex solvers commonly used in machine learning. Similar non-convex optimization problems have been addressed in [19, 20] using the ConCave Convex Programming (CCCP) strategy. According to CCCP strategy, the non-convex cost function $J(\theta)$ is decomposed as the sum of a convex part $J_{vex}(\theta)$ and a concave part $J_{cav}(\theta)$, where θ is the optimization argument. Each iteration of the CCCP procedure approximates the concave part by its tangent and minimizes the resulting convex function (Algorithm 1).

Algorithm 1: ConCave Convex Programming (CCCP)

```
Initialize $\theta^0$
repeat
    $\theta^{t+1} = \arg\min_\theta \; (J_{vex}(\theta) + \frac{\partial J_{cav}(\theta)}{\partial \theta} \cdot \theta)$
until convergence of $\theta$
```

Hence, we propose to apply the CCCP strategy for solving the non-convex optimization formulation (7). Detailed application of the CCCP strategy and the resulting algorithm for solving the U-SVM regression formulation (7) are presented next.

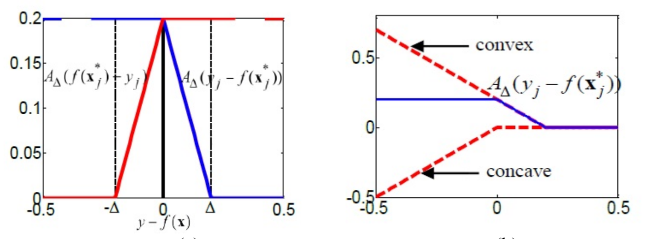
The Universum loss function can be represented as a sum of two ramp losses, $U_\Delta(y_j - f(\mathbf{x}_j^*))$ = $A_\Delta(y_j^* - f(\mathbf{x}_j^*))$ + $A_\Delta(f(\mathbf{x}_j^*) - y_j^*)$ + a constant (see Fig. 3a); where $A_\Delta(t)$ = $\max(0, \Delta - t) - \max(0, -t)$ (see Fig. 3b). The constant term does not affect the optimization; and hence (7) can be re-written as,

$$\min_{\mathbf{w},b,\xi,\xi^*,\zeta,\zeta^*} \; \frac{1}{2}(\mathbf{w}\cdot\mathbf{w}) + \; C\sum_{i=1}^{n}(\xi_i + \xi_i^*) + \; C^*\sum_{j=1}^{m}(\zeta_j + \zeta_j^*) - \sum_{j=1}^{m} H(y_j^*, f(\mathbf{x}_j^*)) \qquad (8)$$

$$\begin{aligned} \text{s.t.} \quad & y_i - (\mathbf{w}\cdot\mathbf{x}_i) - b \le \varepsilon + \xi_i & \xi_i \ge 0 \\ & (\mathbf{w}\cdot\mathbf{x}_i) + b - y_i \le \varepsilon + \xi_i^* & \xi_i^* \ge 0 \\ & y_j^* - (\mathbf{w}\cdot\mathbf{x}_j^*) - b \le -\Delta + \zeta_j & \zeta_j \ge 0 \\ & (\mathbf{w}\cdot\mathbf{x}_j^*) - b - y_j^* \le -\Delta + \zeta_j^* & \zeta_j^* \ge 0 \\ & i = 1\ldots n, \quad j = 1\ldots m \end{aligned}$$

where,

$$H(y_j^*, f(\mathbf{x}_j^*)) = \max(0, -y_j^* + (\mathbf{w}\cdot\mathbf{x}_j^*) + b) + \max(0, y_j^* - (\mathbf{w}\cdot\mathbf{x}_j^*) - b)$$



(a) (b)

Fig. 3. (a) Universum loss as the sum of two ramp losses $A_\Delta(y_j^* - f(\mathbf{x}_j^*))$ and $A_\Delta(f(\mathbf{x}_j^*) - y_j^*)$. (b) Decomposition of $A_\Delta(y_j^* - f(\mathbf{x}_j^*))$ as the sum of a convex and concave loss.

and $f(\mathbf{x}) = (\mathbf{w}\cdot\mathbf{x}) + b$. Define: $k_j = -C^* \frac{\partial H(y_j^*, f(\mathbf{x}_j^*))}{\partial f(\mathbf{x}_j^*)}$

$$\Rightarrow k_j = \begin{cases} -C^* \; ; \; if \; y_j^* < f(\mathbf{x}_j^*) \\ C^* \; ; \; if \; y_j^* > f(\mathbf{x}_j^*) \\ 0 \; ; \quad \text{else} \end{cases}$$

Hence,

$$-C^* \frac{\partial H(y_j^*, f(\mathbf{x}_j^*))}{\partial \theta} \cdot \theta = -C^* \frac{\partial H(y_j^*, f(\mathbf{x}_j^*))}{\partial f(\mathbf{x}_j^*)} \cdot \frac{\partial f(\mathbf{x}_j^*)}{\partial \theta} \cdot \theta$$

$$= k_j \begin{bmatrix} \mathbf{x}_j^* \\ 1 \end{bmatrix}^T \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix} = k_j(\mathbf{w}\cdot\mathbf{x}_j^* + b) \quad \text{with } \theta = \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}.$$

Hence, application of the CCCP strategy to solve formulation (7) yields the following Algorithm 2.

Algorithm 2: CCCP algorithm for U-SVR

```
1. Initialize $(\mathbf{w}^0, b^0)$ using standard SVR model (see eq. (6))
repeat
    2. At $t+1$ iteration update,
       $k_j^{t+1} = \begin{cases} -C^* \; ; \; if \; y_j^* < (\mathbf{w}^t\cdot\mathbf{x}_j^*) + b^t \text{ and } j = 1\ldots m \\ C^* \; ; \; if \; y_j^* > (\mathbf{w}^t\cdot\mathbf{x}_j^*) + b^t \text{ and } j = 1\ldots m \\ 0 \; ; \; \text{else} \end{cases}$
    3. Solve the following eq. (8) to obtain $(\mathbf{w}^{t+1}, b^{t+1})$
       $\min_{\mathbf{w},b,\xi,\xi^*,\zeta,\zeta^*} \frac{1}{2}(\mathbf{w}\cdot\mathbf{w}) + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) + C^*\sum_{j=1}^{m}(\zeta_j + \zeta_j^* - k_j^{t+1}(\mathbf{w}\cdot\mathbf{x}_j^* + b))$
       s.t. $y_i - (\mathbf{w}\cdot\mathbf{x}_i) - b \le \varepsilon + \xi_i \quad \xi_i \ge 0$
            $(\mathbf{w}\cdot\mathbf{x}_i) + b - y_i \le \varepsilon + \xi_i^* \quad \xi_i^* \ge 0$
            $y_j^* - (\mathbf{w}\cdot\mathbf{x}_j^*) - b \le -\Delta + \zeta_j \quad \zeta_j \ge 0$
            $(\mathbf{w}\cdot\mathbf{x}_j^*) - b - y_j^* \le -\Delta + \zeta_j^* \quad \zeta_j^* \ge 0$
            $i = 1\ldots n, \quad j = 1\ldots m$
until convergence i.e. $k_j^{t+1} = k_j^t \quad \forall i = n+1\ldots n+m$
```

The Algorithm 2 can be extended to nonlinear case by transforming the problem to dual form (as shown next). Rewrite,

$$\mathbf{x}_i = \begin{cases} \mathbf{x}_i & i = 1 \ldots n \quad \text{(training samples)} \\ \mathbf{x}_j^* & j = 1 \ldots m \; ; \; i = n+1 \ldots n+m \quad \text{(universum samples)} \end{cases} \qquad (9)$$

$$y_i = \begin{cases} y_i & i = 1 \ldots n \\ y_j^* & j = 1 \ldots m \; ; \; i = n+1 \ldots n+m \end{cases}$$

$$\rho_i = \begin{cases} \varepsilon & i = 1 \ldots n \\ -\Delta & i = n+1 \ldots n+m \end{cases}$$

$$C_i = \begin{cases} C & i = 1 \ldots n \\ C^* & i = n+1 \ldots n+m \end{cases}$$

$$\delta_i = \begin{cases} C^* & \textit{if } \; y_i < f(x_i) \quad \textit{and } \; i = n+1 \ldots n+m \\ 0 & \textit{else} \end{cases}$$

$$\gamma_i = \begin{cases} C^* & \textit{if } \; y_i > f(x_i) \quad \textit{and } \; i = n+1 \ldots n+m \\ 0 & \textit{else} \end{cases}$$

Hence, we obtain the following Algorithm 3 in dual form. The derivation is based on standard KKT conditions and is not shown in this paper.

```
Algorithm 3: CCCP algorithm for U-SVR in dual form
1. Initialize $(\alpha^0, \beta^0, b^0)$ using the standard SVR (in dual form)
repeat
    2. At $t+1$ iteration update,
        $\delta_i^{t+1} = \begin{cases} C^* \; \textit{if} \; y_i < \sum_{i=1}^{n+m} (\alpha_i^t - \beta_i^t)(x_i \cdot x) + b \\ 0 \; \textit{else}, \; i = n+1 \ldots n+m \end{cases}$
        $\gamma_i^{t+1} = \begin{cases} C^* \; \textit{if} \; y_i > \sum_{i=1}^{n+m} (\alpha_i^t - \beta_i^t)(x_i \cdot x) + b \\ 0 \; \textit{else}, \; i = n+1 \ldots n+m \end{cases}$
    3. Solve the following eq. (10) to obtain $(\alpha^{t+1}, \beta^{t+1}, b^{t+1})$
        $\min_{\alpha,\beta} \; \frac{1}{2} \sum_{i,j=1}^{n+m} (\alpha_i - \beta_i)(\alpha_j - \beta_j) K(\mathbf{x}_i \cdot \mathbf{x}_j) + \sum_{i=1}^{n+m} \rho_i (\alpha_i + \beta_i) - \sum_{i=1}^{n+m} y_i (\alpha_i - \beta_i)$
        s.t. $\sum_{i=1}^{n+m} \alpha_i = \sum_{i=1}^{n+m} \beta_i \quad i = 1 \ldots n+m$     (10)
             $-\gamma_i^t \le \alpha_i \le C_i - \gamma_i^t \; ; \; -\delta_i^t \le \beta_i \le C_i - \delta_i^t$
until convergence i.e. $\delta_j^{t+1} = \delta_j^t, \; \gamma_j^{t+1} = \gamma_j^t \quad \forall i = n+1 \ldots n+m$
```

This CCCP based minimization may have local optima, so a good initialization and stopping criteria are critical for this algorithm. In our implementation, standard SVR model (estimated from training samples) is used as the initial condition (for Algorithms 2 & 3). Thus the CCCP strategy searches for local minima near the SVR solution. Further, at each iteration we solve an SVR-like formulation (see eq. 10). The only difference is in the constraints. That is, the dual variables in U-SVR formulation (10) have different upper and lower range values, as compared to standard SVR formulation. Hence, the time complexity for solving the U-SVR formulation using CCCP is similar to solving a standard SVR formulation with $(n+m)$ samples at each iteration. A more detailed analysis on the convergence of the generic CCCP algorithm is available in [21, 22]. Further, analysis shown in [22] guarantees *at least* linear convergence to stationary points for the U-SVR formulation (7). In fact, our own experience indicates fast convergence (2-5 iterations) for many data sets. So this strategy should be scalable for most real-life data sets.

The kernelized version of U-SVR formulation has five tunable parameters: C , C^* kernel parameter, ε and Δ . So model selection (parameter tuning) becomes an issue for real-life applications. In this paper, we propose a simple two-step strategy for U-SVR model selection:

1. Model selection for standard SVR regression. This step performs estimation of standard SVM regression model using only training samples. Following [23], we can select C parameter analytically, e.g. by setting $C = y_{\max} - y_{\min}$, and then perform tuning of ε and kernel parameters via resampling or separate validation data set. These tuning parameters for U-SVR formulation (7) depend only on the training data (not on Universum data).
2. Model selection for tuning two Universum-specific parameters. This step performs selection (or tuning) of parameters C^*/C and Δ specific to the U-SVR formulation, while keeping C, ε and kernel parameters fixed (as obtained in Step 1). This can be performed using a separate validation set or via resampling.

IV. Generating Universum Data and understanding Universum-SVM Regression

Generally, Universum contains data samples from the same application domain as available training data. For example, for handwritten digit recognition application one can use examples of handwritten letters as Universum data samples, along with examples of handwritten digits used as training data [4]. Notably, Universum samples follow a different distribution than the training data.

For many real-life applications, Universum data is readily available. However, selection of good Universum requires application-domain knowledge. An alternative strategy is to generate synthetic Universum directly from available labeled training data – similar to synthetic Universum for classification [3]. Next, we introduce several strategies for generating synthetic Universum under regression setting.

Note that Universum samples for regression are *labeled*, unlike unlabeled Universum samples under classification setting. Hence, under regression setting, the distribution of Universum samples can be different from the distribution of $\mathbf{x}$ - values (of the training data) or their y - values or both. This observation motivates several practical strategies for generating synthetic universum from training samples:

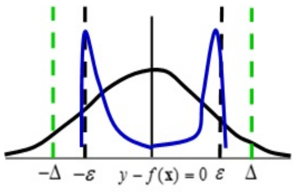


Fig. 4. Representation of the histogram of residuals $y-f(\mathbf{x})$ for regression models. Training samples are shown in blue and Universum samples in black. The estimated $\pm\varepsilon$ and $\pm\Delta$ values are shown in black and green lines respectively.

Strategy 1: keep the marginal distributions of $\mathbf{x}$ and y values fixed, but change the underlying conditional distribution $p(y\,|\,\mathbf{x})$. For example, randomly select any two samples from the training data $(\mathbf{x}_1, y_1)$ and $(\mathbf{x}_2, y_2)$, such that $y_1 \geq \mu_y$ and $y_2 \leq \mu_y$; where μ_y is the mean of the y - values of training samples. Next, permute the samples to create two new universum samples, i.e. $(\mathbf{x}_1, y_2)$ and $(\mathbf{x}_2, y_1)$.

Strategy 2: change the distribution of the y - values of training samples. For example, randomly select a training sample $(\mathbf{x}, y)$ and then replace its y - value as $y' \sim \mathbb{N}(\mu_y, \sigma_y)$ (normal distribution), where μ_y and σ_y are the mean and standard deviation of y - values of the training data.

Each of these strategies modifies the overall distribution of the $(\mathbf{x}, y)$ - values of the training data. So these strategies yield universum data having distribution different from the training data and *should be falsified* by the U-SVR formulation (7). Section V, shows empirical results for the U-SVR method using Universum generated using these strategies. Other strategies that involve changing both $\mathbf{x}$ and y - distributions of training data can be easily designed, but they are not discussed here due to space constrains.

Further, for better understanding of the U-SVR modeling results we adopt the technique known as '*histogram of projections'* originally introduced for SVM classification setting [17, 18, 24]. Under classification setting, the 'projection value' for a given input measures its distance from SVM decision boundary. For regression, a similar quantity is the residual $y-f(\mathbf{x})$ that measures the difference between response y and its estimate $f(\mathbf{x})$. So for regression models we use the univariate *histogram of residuals* $y-f(\mathbf{x})$, where $f(\mathbf{x})$ is the trained regression model. A typical histogram of residuals for trained SVR model is shown in Fig. 4. In addition, Fig. 4 also shows projections of the residual values $y^*-f(\mathbf{x}^*)$ of universum samples. Similar to methodology developed for U- SVM classification, visual interpretation of the histograms of residuals for training data and Universum data can be used for understanding the effectiveness of Universum under regression setting. In particular, Fig. 4 shows the effect of data piling or clustering of residual values for training samples near the $\pm\varepsilon$ boundaries, which is similar to data piling at the margin borders for SVM classification [17, 18, 24]. This data piling effect is typically observed for 'small-sample' regression data sets corresponding to very ill-posed estimation problems [25]. For such ill-posed settings, introducing additional constraints (in the form of Universum data) usually improves the quality of the estimated models. For example, assuming the distribution of residuals for Universum samples (relative to standard SVR model) is as shown in Fig. 4, application of U-SVR is expected to modify/improve the original SVR model by pushing the Universum samples further away from a regression model, according to Universum loss function (shown in Fig. 2b).

V. Empirical Results

A. Synthetic Hypercube dataset

Our *first experiment* uses a synthetic 30-dimensional hypercube data set, where each input is uniformly distributed in [0, 1] interval. The output is generated as:

$$y = \quad x_1 + \ldots + x_5 - x_6 - \ldots - x_{10} + \ldots - x_{30} + \qquad \delta$$

where, the noise is Gaussian: $\delta \sim \mathbb{N}(0, \sigma\mathbf{I})$. For this data set, we use linear SVR parameterization and consider two different types of universum.

Universum 1: input samples follow the same distribution as training samples, e.g., $\mathbf{x} \in \mathbb{R}^{30}$ and uniformly distributed in [0, 1]. The output is generated by changing the sign i.e.,

$$y = \quad -x_1 - \ldots - x_5 + x_6 + \ldots + x_{10} - \ldots + x_{30}$$

Universum 2: following strategy 2 for generating synthetic universum, we randomly select a training sample $(\mathbf{x}, y)$ and re-set its y - value as $y' \sim \mathbb{N}(\mu_y, \sigma_y)$ (normal distribution), where μ_y and σ_y are the mean and standard deviation of y - values of the training data.

The experimental setting is specified below:

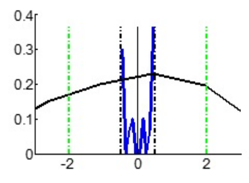
– No. of training and validation samples = 30, 150 (characterizing low and high sample-size settings, respectively. The number of validation samples is always set to be the same as the number of training samples. This independent validation set is used for model selection).

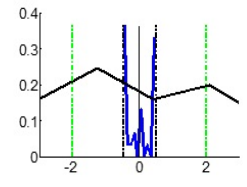
– No. of test samples = 5000.

– No. of universum samples = 300.

We consider two additive noise levels σ = 0.5 and 2, in order to capture the effects of *low* and *high* noise levels respectively. For the high sample size setting we provide the results for σ = 0.5 only. Experiments involving high noise conditions for high sample size settings did not yield any additional improvement when using the U-SVR formulation.

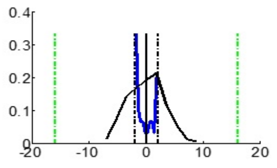
Performance results in Table 1 show the average *Normalized Root Mean Squared* (*NRMS* = $\dfrac{\sqrt{(1/n)\sum_{i=1}^{n}(y-\hat{y})^2}}{std(y)}$) error for training and test data observed over 25 random experiments. Here, n = no. of samples, and $\hat{y}$

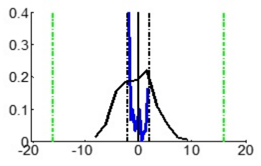




(a) (b)

Fig. 5. Histogram of residuals for training samples (in blue) and Universum 1 samples (in black) with no. of training samples n = 30 and σ = 0.5. (a) histogram for standard SVR model (C = 6.73, ε = 0.5) (b) histogram for U-SVR model (C^*/C = 0.05, Δ = 2).



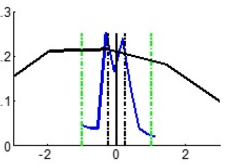


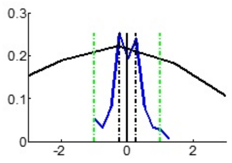
(a) (b)

Fig. 6. Histogram of residuals for training samples (in blue) and Universum 1 samples (in black) with no. of training samples n = 30 and σ = 2. (a) histogram for standard SVR model (C = 14.5, ε = 2) (b) histogram for U-SVR model (C^*/C = 0.001, Δ = 16).

denotes estimated output values. For each experiment we randomly select the training/validation/test set. The standard deviation of the NRMS values (over 25 experiments) is shown in parenthesis. These empirical results in Table I indicate that for low-sample settings introducing Universum can indeed improve prediction performance, especially for low-noise settings. However, for high-sample size settings introducing Universum does not yield any improvement relative to standard SVR or ridge regression.

These modeling results for U-SVR can be understood using the 'histogram-of-residuals' technique described in Section 4. Actual histograms of residuals for the synthetic hypercube dataset are shown in Figs 5, 6 and 7. Figs. 5(a)-(b) show the histograms for the estimated SVR and U-SVR models under low sample size, low noise settings. The histogram clearly shows the effect of data piling for training samples, and also shows that the distribution for Universum 1 data is unimodal and centered around the SVR model (marked as the point '0' on x-axis in Fig. 5a). Therefore, we can expect that introducing Universum 1 will change/improve the regression model. This is confirmed by analyzing the histograms of residuals for the trained U-SVR shown in Fig. 5b, which shows the effect of pushing Universum 1 samples away from the estimated regression model. Specifically, for the SVR model (in Fig. 5a), the fraction of universum samples lying within the $\pm\Delta$ zone is ~ 92% and that for the U-SVR model (in Fig. 5b) is ~ 85%. Hence, the U-SVR model increases the number of contradictions for the universum samples and improves the prediction performance (relative to standard SVR), which is consistent with results in Table I. Next, we analyze histograms of residuals for the low sample size, high noise (n =30, σ =2) data shown in Figs. 6(a)-(b). In this case, the data piling effect for standard SVR model is less strong (as compared to low-noise setting in Figs. 5(a). Further, visual comparison of the





(a) (b)

Fig. 7. Histogram of residuals for training samples (in blue) and Universum 1 samples (in black) with no. of training samples n = 150 and σ = 0.5. (a) histogram for standard SVR model (C = 9.65, ε = 0.25) (b) histogram for U-SVR model (C^*/C = 0.5, Δ = 1).

TABLE I. COMPARISON OF TEST ERROR FOR DIFFERENT UNIVERSUM.

NRMS (in %)	Ridge Regression	SVR	U-SVR (type 1)	U-SVR (type 2)
low sample size ($n = 30$), low noise ($\sigma = 0.5$)				
Test	54.5 (9.3)	55.3 (9.1)	47.7 (7.9)	53.7 (9.6)
Training	21.3 (9.3)	20.5 (9.5)	20.2 (8.7)	20.3 (8.9)
low sample size ($n = 30$), high noise ($\sigma = 2$)				
Test	97.1 (10.2)	97.6 (11.8)	92.6 (16.3)	97.4 (13.9)
Training	74.1 (17.8)	79.4 (17.1)	76.5 (12.9)	79.9 (23.4)
high sample size ($n = 150$), low noise ($\sigma = 0.5$)				
Test	15.8 (0.8)	16.2 (2.1)	16.4 (2.1)	15.9 (2.2)
Training	14.7 (1.6)	14.6 (1.3)	14.9 (1.3)	14.4 (1.4)

histograms for the SVR and U-SVR models suggests no significant change in the fraction of Universum samples within the $\pm\Delta$ zone. Hence, we can expect only minor or no improvement in the prediction performance for U-SVR, which is confirmed by results in Table I. Finally, consider large sample size, low noise settings (n = 150, σ = 0.5). Having large number of training samples yields very accurate SVR models. For such data sets, we do not observe the data piling effect at the $\pm\varepsilon$ values (see Figs. 7(a)-(b)), so we expect no improvement from introducing Universum data (see Table I). These experiments show that, U-SVR is particularly effective for very sparse settings (~ high-dimensional and low sample size) under low noise conditions. Under such settings, the training data exhibits large data-piling near the $\pm\varepsilon$ margins for the SVR model, and introducing the Universum usually helps to improve the prediction performance. With increased noise level, the data-piling effect decreases and the U-SVR yields no improvement over the SVR solution. Finally, when the number of training samples is large, the estimation problem becomes well-posed and SVR model does not exhibit any data-piling at the $\pm\varepsilon$ margins. In this case, application of U-SVR does not provide any improvement. The histograms for Universum 2 do not provide additional insights, and have been omitted from this paper.

The next experiment follows the same experimental set up for 30-dimensional hypercube data set under low-sample size setting as above, except that the additive noise level is set to zero (σ = 0). Based on our previous discussion, this data set is expected to show the most significant improvement in prediction performance of U-SVR (relative to standard SVR). Table II shows performance comparisons, suggesting that U-SVR (using Universum 1) indeed provides very significant improvement over standard SVR solution.

TABLE II. COMPARISON OF AVERAGE TEST ERROR FOR DIFFERENT UNIVERSUM WITH ($n = 30$) AND ZERO NOISE.

NRMS (in %)	**Ridge Regression**	**SVR**	**U-SVR** (type 1)	**U-SVR** (type 2)
Test	15.9 (10.3)	17.5 (10.2)	7.3 (6.9)	17.4 (10.2)
Training	0.15 (0.01)	1.6 (2.4)	0.4 (1.0)	1.2 (1.6)

TABLE III. COMPARISON OF TEST ERROR WITH VARYING UNIVERSUM SAMPLE SIZE. NRMS (IN %) FOR SVR = 56.98 (9.06).

NRMS	**Number of Universum samples**			
(in %)	m=50	m=100	m=300	m=500
U-SVR (type1)	55.3 (8.5)	52.7 (6.5)	47.1 (9.0)	47.1 (9.1)
U-SVR (type2)	55.0 (11.1)	56.7 (9.8)	56.6 (9.1)	56.3 (9.1)

Finally, our last set of experiments demonstrates the generalization performance of U-SVR with increase in universum data samples. We use the same synthetic hypercube data set under small sample size low noise setting, and vary the Universum sample size (shown in Table III). These results indicate that for sufficiently large number of universum samples the effectiveness of U-SVR depends mostly on the *type* (~ statistical characteristics) of the universum data. Similar to classification settings [4, 5], the effectiveness of Universum for regression problems depends on the statistical characteristics of both the training and the universum data sets. Hence, there is a need for additional research aimed at better understanding and statistical characterization of '*good'* universum data sets.

B. Real world datasets

Next we provide empirical results on several publicly available real-world datasets. The datasets include,

Computer Hardware (CPU) [26]: The goal for this dataset is to predict the *published relative CPU performance* using several other CPU properties. Here, the categorical variable *vendor_name* has been transformed to a binary representation. Further, the y - values have been scaled as log(1 + y).

Diesel dataset [27]: The goal here is to predict the total aromatics of the fuel sample using the NIR spectra of diesel fuels ranging from 750 to 1550 nm, discretized into 401 wavelength values. The dataset has 784 samples of which 389 samples have missing measurements for total aromatics. In this experiment we remove the samples with missing values.

Kelly Blue Book (KBB) dataset [28]: The goal here is to predict the retail price of used 2005 GM cars based on a variety of characteristics such as mileage, make, model, engine size, interior style, and cruise control etc. Here, the categorical variables *make, model, trim* have been transformed to a binary representation. Further, the y - values have been scaled as log(1 + y).

Vilmann's Rat dataset [29]: The dataset contains the skull X-ray images of 21 different rats, represented using 8 - landmarks for 2-dimensions. The skull X-ray images are available for each rat at ages: 7, 14, 21, 30, 40, 60, 90 and 150 days i.e. a total of (21 x 8) = 168 samples. The task is to predict the ontogenetic development (age) of a rat using the X-ray images. The dataset contains 4 missing data which have

TABLE IV. EXPERIMENT SETTINGS FOR REAL WORLD DATASETS.

Dataset	# training/ validation	# test	# universum	dimension
CPU	50/50	109	200	36
Diesel	100/100	195	200	401
KBB	200/200	404	200	97
Rat	40/40	84	200	16

TABLE V. RESULTS ON REAL WORLD DATASETS. THE 1ST AND 2ND ROWS PROVIDES NRMS (IN %) AND TYPICAL PARAMETERS RESPECTIVELY.

Data	**SVR**	**U-SVR (type1)**	**U-SVR (type2)**
CPU	**55.01 (7.24)** [C ~ 4.5 , ε =0.1,1]	**52.2 (5.8)** [$\frac{C^*}{C}$ = 2^{-4}-2^0, Δ = 2^{-2}-2^0]	**54.2 (6.6)** [$\frac{C^*}{C}$ = 2^{-6}, Δ = 2^{-1}]
Diesel	**26.1 (12.3)** [C ~ 31, ε = 2^{-1}]	**24.1 (9.7)** [$\frac{C^*}{C}$ = 2, Δ = 2^2 -2^4]	**22.03 (8.7)** [$\frac{C^*}{C}$ = 2, Δ = 1,4]
KBB	**10.6 (2.4)** [C ~ 2, ε = 0.01]	**9.47 (1.4)** [$\frac{C^*}{C}$ = 2^{-6}, Δ =0.01-1]	**9.48 (1.4)** [$\frac{C^*}{C}$ = 2^{-6}, Δ = 0.05 -0.8]
Rat	**26.1 (3.1)** [C ~ 143, ε = 4, 8]	**25.7 (2.6)** [$\frac{C^*}{C}$ = 2^{-3} - 2^{-1}, Δ = 2^5 - 2^6]	**24.8 (2.7)** [$\frac{C^*}{C}$ = 0.5, Δ = 2^5 - 2^6]

been removed from the analysis. Finally, the $\mathbf{x}$ -values of the training data have been pre-scaled uniformly to the same range [-1, 1] for all datasets. This pre-processing is necessary for all SVM-based methods [18].

For our experiments, we use two types of synthetic universum generated from the training data, i.e. following Strategy 1 (Universum 1) and Strategy 2 (Universum 2) as described in Section IV. The number of universum samples is selected to be sufficiently large, so that additional increase in universum samples does not provide any improvement. Table IV provides detailed description of the experimental setting used. Further, our initial experiments suggest that linear SVM parameterization is appropriate for the first three datasets. However, for the *Rat* dataset we use an RBF kernel of the form $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|_2^2)$ with $\gamma = 2^{-3} - 2^{-2}$.

Performance comparisons between SVR and U-SVR models over 10 random partitioning of the data into training, validation and test data sets are shown in Table V. Table V provides the average NRMS values for test data sets. The standard deviation is shown in parenthesis. Further, Table V also provides the typical optimal SVR and U-SVR model parameters. These parameters are selected using the two-step strategy described in section III B, where tuning parameters are selected using separate validation data (see Table IV). These modeling results suggest that U-SVR provides improved generalization over standard SVR. However, the level of improvement depends on the characteristics of the training data and the Universum types. The effectiveness of introducing Universum data can be analyzed using the 'histogram-of-residuals' technique that shows the distribution of the Universum data relative to the distribution of residuals of training samples. This analysis is omitted due to space constraints.

VI. Conclusion

This paper extends the idea of universum learning to regression problems and provides new Universum Support Vector Regression (U-SVR) formulation. This U-SVR formulation is non-convex and cannot be solved using standard convex solvers. This paper adopts the method of ConCave Convex Problem (CCCP) and provides a new algorithm (Algorithm 1 & 2) for solving the proposed U-SVR formulation. Following this strategy, the current U-SVR formulation can be solved by several iterations of standard SVM-like optimization problem.

Moreover, the proposed U-SVR formulation has 5 tunable parameters: C , C^* , ε , Δ and kernel parameter. Hence a successful practical application of U-SVR depends on the optimal selection of these model parameters. We propose simple two-step strategy for model selection where optimal model parameters for standard SVR are estimated first, and then model selection for U-SVR involves tuning only two remaining parameters C^*/C and Δ . Such a two-step strategy significantly simplifies model selection for U-SVR.

Finally, the paper provides empirical results to show the effectiveness of the proposed U-SVR over standard SVR. Our results suggest that U-SVR is particularly effective for high-dimension low (training) sample size settings. Under such settings, the SVR model exhibits significant data piling of the training samples near the $\pm\varepsilon$ margin. For such ill-posed settings, introducing the Universum can provide improved generalization over the standard SVR solution. However, the effectiveness of U-SVR depends on the statistical characteristics of both the training and Universum data. These statistical characteristics can be conveniently captured using the `histogram-of-residuals' method introduced in this paper.

Acknowledgement: The work of V. Cherkassky was supported, in part, by NIH grant U01NS073557 and by NIH grant R01NS092882.

References

[1] V. Vapnik. Estimation of Dependences Based on Empirical Data (Information Science and Statistics). Springer, Mar. 2006.

[2] V. N. Vapnik. Statistical Learning Theory. Wiley-Interscience, 1998.

[3] J. Weston, R. Collobert, F. Sinz, L. Bottou, and V. Vapnik. Inference with the universum. ICML, pages 1009-1016, 2006.

[4] V. Cherkassky, S. Dhar, and W. Dai. Practical conditions for effectiveness of the universum learning. IEEE Transactions on Neural Networks, 22(8):1241-1255, 2011.

[5] F. Sinz, O. Chapelle, A. Agarwal, and B. Scholkopf. An analysis of inference with the universum. NIPS , pp 1369-1376, NY, Sept. 2008.

[6] S. Chen and C. Zhang. Selecting informative universum sample for semi-supervised learning. In IJCAI, pages 1016-1021, 2009.

[7] S. Dhar and V. Cherkassky. Development and evaluation of cost-sensitive universum-svm. IEEE Transactions on Cybernetics, 45(4):806-818, 2015.

[8] S. Lu and L. Tong. Weighted twin support vector machine with universum. Advances in Computer Science: an International Journal, 3(2):17-23, 2014.

[9] Z. Qi, Y. Tian, and Y. Shi. A nonparallel support vector machine for a classication problem with universum learning. Journal of Computational and Applied Mathematics, 263:288- 298, 2014.

[10] C. Shen, P. Wang, F. Shen, and H. Wang. U-boost: Boosting with the universum, IEEE Transactions on Pattern Analysis and Machine Intelligence,34(4):825-832, 2012.

[11] Z. Wang, Y. Zhu, W. Liu, Z. Chen, and D. Gao. Multi-view learning with universum. Knowledge-Based Systems, 70:376-391, 2014.

[12] D. Zhang, J. Wang, F. Wang, and C. Zhang. Semi-supervised classication with universum. SDM, pages 323-333, 2008.

[13] C. Zhu. Improved multi-kernel classification machine with nystrom approximation technique and universum data. Neurocomputing 175:610–634, 2016.

[14] S. Dhar , N. Ramakrishnan,V. Cherkassky, M. Shah. Universum Learning for Multiclass SVM. arXiv preprint arXiv:1609.09162. 2016.

[15] S. Dhar, Analysis and extensions of Universum learning, Ph.D. dissertation, ECE, UMN, MN, 2014.

[16] S. Dhar, V. Cherkassky, Universum Learning for SVM Regression, arXiv preprint arXiv:1605.08497, 2016.

[17] V. Cherkassky. Predictive Learning. VCtextbook, 2013.

[18] V. Cherkassky and F. M. Mulier. Learning from Data: Concepts, Theory, and Methods. Wiley-IEEE Press, 2007.

[19] X. Shen, G. C. Tseng, X. Zhang, and W. H. Wong. On psi-Learning. Journal of the American Statistical Association, 98:724-734, Jan. 2003.

[20] R. Collobert, F. Sinz, J. Weston, and L. Bottou. Large scale transductive svms. The Journal of Machine Learning Research, 7:1687-1712, 2006.

[21] R. Lanckriet, and B. Sriperumbudur. On the convergence of the concave-convex procedure. NIPS, 2009.

[22] I. Yen, N. Peng, P. Wang & S. Lin. On convergence rate of concave-convex procedure. NIPS Optimization Workshop, 2012.

[23] V. Cherkassky and Y. Ma. Practical selection of svm parameters and noise estimation for svm regression. Neural networks, 17(1), 2004.

[24] V. Cherkassky and S. Dhar. Simple method for interpretation of high-dimensional nonlinear SVM classication models. DMIN, 2010.

[25] S. Boyd and L. Vandenberghe. Convex Optimization. Cambridge University Press, New York, NY, USA, 2004.

[26] D. Kibler, D. Aha, and M. Albert, "Instance based prediction of real-valued attributes." Computational Intelligence 5(2):51–57.

[27] EigenvectorResearch. 2016. Online : http://www.eigenvector.com/data/swri/.

[28] S. Kuiper. Introduction to multiple regression: How much is your car worth?. Journal of Statistics Education 16(3), 2008

[29] F. L. Bookstein. Morphometric tools for landmark data: geometry and biology. Cambridge University Press, 1997.