# A New Learning Paradigm for Random Vector Functional–Link Network: RVFL+

**Article** · August 2017

**1 author:**

Pengbo Zhang
The Hong Kong University of Science and Technology
**10** PUBLICATIONS **152** CITATIONS

# A New Learning Paradigm for Random Vector Functional-Link Network: RVFL+

Peng-Bo Zhang

*Abstract*—In school, a teacher plays an important role in various classroom teaching patterns. Likewise to this human learning activity, the learning using privileged information (LUPI) paradigm provides additional information generated by the teacher to 'teach' learning algorithms during the training stage. Therefore, this novel learning paradigm is a typical *Teacher-Student Interaction* mechanism. This paper is the first to present a random vector functional link network based on the LUPI paradigm, called RVFL+. Rather than simply combining two existing approaches, the newly-derived RVFL+ fills the gap between neural networks and the LUPI paradigm, which offers an alternative way to train RVFL networks. Moreover, the proposed RVFL+ can perform in conjunction with the kernel trick for highly complicated nonlinear feature learning, which is termed KRVFL+. Furthermore, the statistical property of the proposed RVFL+ is investigated, and we derive a sharp and high-quality generalization error bound based on the Rademacher complexity. Competitive experimental results on 14 real-world datasets illustrate the great effectiveness and efficiency of the novel RVFL+ and KRVFL+, which can achieve better generalization performance than state-of-the-art algorithms.

*Index Terms*—RVFL+, KRVFL+, Learning using privileged information, the Rademacher complexity, SVM+, Random vector functional link networks.

## I. INTRODUCTION

RECENTLY, Vapnik and Vashist [1] provided a new learning paradigm termed learning using privileged information (LUPI), which is aimed at enhancing the generalization performance of learning algorithms. Generally speaking, in classical supervised learning paradigm, the training data and test data must come from the same distribution. Although in this new learning paradigm the training data is also considered an unbiased representation for the test data, the LUPI provides a set of additional information for the training data during the training stage, which is called privileged information. In the LUPI paradigm, we use the new training set containing privileged information to train a learning algorithm, while the privileged information is not available in the test stage. We note that the new learning paradigm is analogous to human learning process. In class, a teacher can provide some important and helpful information about this course for students, and these information provided by the teacher can help students acquire knowledge better. Therefore, a teacher plays an essential role in human leaning process. The LUPI paradigm resembling the classroom teaching model can achieve better generalization performance than the traditional learning paradigm.

The author is with Department of Industrial Engineering and Logistics Management, School of Engineering, Hong Kong University of Science and Technology, Hong Kong 999077, China.(Email: pzhangag@connect.ust.hk.).

Vapnik and Vashist [1] was the first to derive a SVM algorithm with privileged information termed SVM+, which leverages the strength of the LUPI paradigm. A thorough theoretical analysis of the SVM+ was further illustrated in [2], [3]. Previous work of the LUPI paradigm focus on two aspects: solving the LUPI based algorithms efficiently and incorporating the LUPI paradigm into various learning algorithms. This paper focuses on the later. The newly-derived RFVL+, however, has simpler optimization constraints than the SVM+. As a result, we can calculate a closed-form solution to the RFVL+, which naturally tackles the former.

From the optimization perspective, the formulation of the SVM+ is a typical quadratic programming (QP) problem, and in general the QP problem can be solved by some optimization toolboxes (for example the CVX toolbox [4]). However, it is unnatural and inconvenience to train a learning algorithm by some optimization toolboxes in real-world applications. For this reason, it is necessary to derive an efficient approach to solve it [5]–[7]. Pechyony et. al. [5] presented an SMO-style optimization approach for the SVM+. Li et. al. [6] further derived two fast algorithms for linear and kernel SVM+, respectively. In addition to solving the SVM+ efficiently, the LUPI paradigm is incorporated into various learning algorithms [8]–[12]. Feyereisl et. al. [8] presented a novel structured SVM for object localization, which uses attributes and segmentation masks of an object as privileged information. Fouad et. al. [9] provided a generalized matrix LVQ (GMLVQ) approach based on the LUPI paradigm. In order to tackle the face verification and person re-identification problems better, Xu et. al. [10] used the depth information of RGB-D images as privileged information to derive a novel distance metric learning algorithm. These existing work have confirmed the great effectiveness and generality of various LUPI-based learning algorithms.

Nowadays, neural network is one of the most popular learning algorithms due to the wave of deep learning, and most of current deep learning methods are neural networks, including denoising auto-encoders (DAE) [13], convolutional neural networks (CNNs) [14], deep belief networks (DBNs) [15] and long short-term memory (LSTM) [16], etc. These neural network methods have achieved greatly successes in various real-world applications, including image classification and segmentation, speech recognition, natural language processing, etc. Therefore, it is interesting to combine neural networks with the LUPI paradigm, and to leverage the strengths of neural networks and the LUPI paradigm. The goal of this paper is to tackle this open problem and construct a bridge to link the LUPI paradigm and neural networks.

In this paper, we derive a novel random vector functional link network with privileged information called RVFL+. The random vector functional link network (RVFL) [17]–[20] is a classical single layer feedforward neural network (SLFN), which overcomes some limitations of SLFNs including slow convergence, over-fitting and trapping in a local minimum. Although the RVFL has achieved good generalization performance in some real-world tasks [21], in order to improve further its effectiveness, we provide the LUPI paradigm for the RVFL. Different from existing variants of RVFL, the RFVL+ may open a door towards alternative to the traditional learning paradigm for the RVFL in real-world tasks. In other words, rather than using only the labeled training data in the training stage, the RVFL+ makes use of not only the labeled training data but also a set of the additional privileged information to train the RVFL, which interprets the essential difference between the two learning paradigms.

Moreover, following the kernel ridge regression [22], [23], we further derive a kernel-based RVFL+ called KRVFL+ in order to handle highly complicated nonlinear relationships between high-dimensional data. The KRVFL+ has two major advantages over the RVFL+. On one hand, the random affine transform leading to unpredictability is eliminated in the KRVFL+. Instead, both the original and privileged features are mapped into a reproducing kernel Hilbert space (RKHS). On the other hand, the KRVFL+ no longer considers the number of enhancement nodes, which is a key factor to affect its generalization ability. As a result, the performance of the KRVFL+ in terms of effectiveness and stability is significantly improved in most real-world tasks.

Furthermore, we investigate the statistical property of the newly-derived RVFL+. We provide a tight generalization error bound based on the Rademacher complexity [24] for the RVFL+. Our generalization error bound benefits from the advantageous properties of the Rademacher complexity. The Rademacher complexity is a commonly-used powerful tool to measure the richness of a class of real-valued functions in terms of its inputs, and thus better capture the properties of distribution that generates the date. In the RVFL+, the weights and biases between the input layer and enhancement nodes are generated randomly and are fixed, the output weights are then calculated by the Moore-Penrose pseudo-inverse [17], [20] or the ridge regression [25]. Therefore, the RVFL+ is considered as a 'special' linear prediction. The Rademacher complexity is an ideal choice for the analysis of this type of algorithms, and can provide a high-quality generalization error bound in terms of its inputs. In contrasts to previous work [26], we provide a more tight and general test error bound, and the novel bound also applies to various forms of the RVFL including the newly-derived KRVFL+.

The *contributions* of this paper are summarized as follows.

- We derive a novel random vector functional link network with privileged information, called RVFL+. To the best of our knowledge, the RVFL+ is the first to bridge the gap between neural networks and the LUPI paradigm. Different from existing variants of the RVFL, the newly-derived RVFL+ provides an alternative paradigm to train the RVFL in order to enhance further the generalization performance of the RVFL.

- We extend the RVFL+ to the kernel version called KRVFL+, and the KRVFL+ enables handling effectively highly nonlinear relationships between high-dimensional data.

- The previous work focus on two aspects: deriving an efficient solver for the LUPI-based approaches and combining the LUPI paradigm with various learning algorithms. This paper focus on the latter. However, from the optimization perspective, we find that the proposed RVFL+ has sampler constraints than the SVM+. As a result, we can obtain a closed-form solution to the RVFL+, which naturally tackles the former.

- In this paper, we also provide a theoretical guarantee for the proposed RVFL+. According to the Rademacher complexity, we derive a tight input-dependent test error bound for the RVFL+, and the new generalization error bound also applies to various RVFLs including the proposed KRVFL+.

The remainder of this paper is organized as follows. We brief the related work of the RVFL in Section II. In Section III, we introduce the newly-derived RVFL+ and KRVFL+. We study the statistical property of the RVFL+ and provide a novel tight generalization error bound based on the Rademacher complexity in Section IV. In Section V, we conduct several experiments on 14 real-world datasets to evaluate the proposed RVFL+ and KRVFL+. This paper concludes in Section VI.

## II. RELATED WORK OF RANDOM VECTOR FUNCTIONAL-LINK NETWORKS

Over the last three decades, randomization based algorithms, including random projection [27], random forests [28], bagging [29], stochastic configuration networks (SCN) [30], [31], and random vector functional link networks (RVFL) [19], etc., play important roles in machine learning community. We refer to [32], [33] for great surveys of the randomized neural networks.

The RVFL presented first by Pao et. al. [19] is one of most popular single layer feedforward neural networks due to its universal approximation ability and great generalization performance. Many researchers have investigated numerous variants of the RVFL in various domains during these three decades. Chen and Wan [34] presented two novel algorithms for the functional-link network in order to calculate efficiently the optimal weights and update the weights on-the-fly, receptively. Chen [35] presented a novel single-hidden layer neural network structure, which can rapidly calculate the optimal weights. A new RVFL was presented by Patra et. al. [36] for nonlinear dynamic systems. In addition to these early studies, the RVFL has gained more attention from more and more researchers in recent years. Scardapane et. al. [37] presented a distributed RVFL algorithm in order to improve the efficiency of the RVFL. Scardapane et. al. [38] derived a new RVFL with full Bayesian inference for robust data modeling. Ren et. al. [39] used the RVFL to forecast the short-term electricity load. Xu et. al. [26] built an effective spatiotemporal

model based on a kernel-based RVFL algorithm to forecast the distribution of the temperature. Following the basic idea of the RVFL, Chen and Liu [40] proposed a broad learning system for dealing with the high-volume and time-variety data [41], which provides an alternative way to design the architecture of learning algorithms in this big data era.

## III. RANDOM VECTOR FUNCTIONAL-LINK NETWORKS WITH PRIVILEGED INFORMATION

### A. Preliminaries

The RVFL network is a classical single layer feedforward neural network, and the architecture of the RVFL is shown in Figure 1. The RVFL initializes randomly all weights and biases between the input layer and enhancement nodes, and then these parameters are fixed and do not need to be tuned during the training stage. The output weights on red solid lines in Figure 1 can be calculated by the Moore-Penrose pseudo-inverse [17], [20] or the ridge regression [25]. Moreover, the direct link between the input layer and the output layer is an effective and simple regularization technique preventing RVFL networks from overfitting.
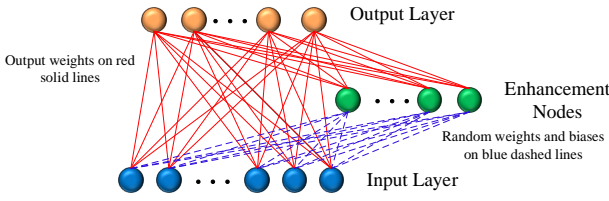


Fig. 1: The architecture of the RVFL network

Given a set of labeled data $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)|\boldsymbol{x}_i \in \mathbb{R}^n, \boldsymbol{y}_i \in \mathbb{R}^m, i = 1, \ldots, N\}$, a RVFL network with $P$ enhancement nodes can be formulated as

$$\boldsymbol{H}\boldsymbol{w} = \boldsymbol{Y} \quad (1)$$

where $\boldsymbol{w}$ is an output weight vector, $\boldsymbol{H}$ is a concatenated matrix combining input data and outputs from the enhancement nodes and $\boldsymbol{Y}$ is a label matrix. $\boldsymbol{H}$ and $\boldsymbol{w}$ are shown as

$$\boldsymbol{H} = [\boldsymbol{H}_1 \quad \boldsymbol{H}_2]$$

$$\boldsymbol{H}_1 = \begin{bmatrix} \boldsymbol{x}_{11} & \cdots & \boldsymbol{x}_{1n} \\ \vdots & \ddots & \vdots \\ \boldsymbol{x}_{N1} & \cdots & \boldsymbol{x}_{Nn} \end{bmatrix}$$

$$\boldsymbol{H}_2 = \begin{bmatrix} G(\boldsymbol{a}_1 \cdot \boldsymbol{x}_1 + b_1) & \cdots & G(\boldsymbol{a}_P \cdot \boldsymbol{x}_1 + b_P) \\ \vdots & \ddots & \vdots \\ G(\boldsymbol{a}_1 \cdot \boldsymbol{a}_N + b_1) & \cdots & G(\boldsymbol{a}_P \cdot \boldsymbol{a}_N + b_P) \end{bmatrix} \quad (2)$$

$$\boldsymbol{w} = \begin{bmatrix} \boldsymbol{w}_1^T \\ \vdots \\ \boldsymbol{w}_{(n+P)}^T \end{bmatrix} \quad (3)$$

In (2), $\boldsymbol{a}_j$ and $b_j$ $(j = 1, \ldots, P)$ are the weight and bias between the input layer and enhancement nodes. According to [21], all weights and biases are from a uniform distribution within $[-u, u]$ and $[0, u]$, respectively, where $u$ is a positive user-defined parameter. This reasonable initial strategy can overcome the limitation that the RVFL with random weights and biases uniformly chosen from $[-1, 1]$ and $[0, 1]$ respectively may not obtain the most optimal solution. $G(\cdot)$ is a nonlinear activation function such as sigmoid, tanh, rbf, etc.

From (1), we can calculate directly the output weights $\boldsymbol{w}$ by the Moore-Penrose pseudo-inverse [17], [20] or the ridge regression [25], which are shown in (4) and (5) respectively.

$$\boldsymbol{w} = \boldsymbol{H}^{\dagger}\boldsymbol{Y} \quad (4)$$

$$\boldsymbol{w} = (\boldsymbol{H}^T\boldsymbol{H} + \frac{\boldsymbol{I}}{C})^{-1}\boldsymbol{H}^T\boldsymbol{Y} \quad (5)$$

where $\dagger$ is the Moore-Penrose pseudo-inverse, $\boldsymbol{I}$ is an identity matrix and $C$ is a trading-off parameter.

The model structure of the RVFL is so simple, why does the RVFL work well for most tasks? Giryes et. al. [42] give a possible theoretical explanation for this open problem. To provide this explanation, first of all, Giryes et. al. reveal the essence of training in learning algorithms. Generally speaking, the angles of samples between different classes are larger than that of samples within the same class [43]. Therefore, the role of training in learning algorithms is that '*the angles between points from different classes are penalized more than the angles between the points in the same class*' [42]. Moreover, in this big data era, due to the highly complicated model architecture, it is quick difficult to tune all parameters in leaning process, which needs extremely high computational cost. To tackle this problem, randomization is an ideal choice for some learning algorithms, leading to cheaper computational cost. A great random initialization allows the learning algorithm to be a universal one before training. Now we revisit the RVFL. The RVFL uses the hybrid strategy to train the entire network. In the RVFL, the random initial parameters between the input layer and the enhancement nodes handle well input samples having distinguishable angles, and the turned output weights further deal with the remaining data.

### B. RVFL+ for Baniry, Multiclass Classification and Regression

Bartlett [44] illustrates that for feedforward neural networks having small empirical error, smaller the norm of weights implies greater generalization performance. From the optimization perspective, we note that the basic idea of the RVFL network is to minimize the training error $\|\boldsymbol{Y} - \boldsymbol{H}\boldsymbol{w}\|_2^2$ and the output weights $\|\boldsymbol{w}\|_2^2$ simultaneously, which emphasizes the combination of the least square loss function and the Tikhonov regularization. As a consequence, the RVFL can achieve a great generalization performance. In order to incorporate the LUPI paradigm, following the dual version of the ridge regression approach [23], we formulate the RVFL as

$$\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \frac{C}{2}\sum_{i=1}^{N}\boldsymbol{\zeta}_i^2$$

$$s.t. \quad \boldsymbol{h}(\boldsymbol{x}_i)\boldsymbol{w} = \boldsymbol{y}_i - \boldsymbol{\zeta}_i, \forall 1 \le i \le N. \quad (6)$$

where $\boldsymbol{h}(\boldsymbol{x}_i)$ is a 'combined' features vector, including the original input features vector $\boldsymbol{h}_1$ and the output features vector

$h_2$ from the enhancement nodes. We define $\boldsymbol{h}(\boldsymbol{x}_i)$ as the enhanced layer output vector. $\boldsymbol{\zeta}_i = [\zeta_{i1}, \ldots, \zeta_{im}]^T$ is a training error vector with $m$ output nodes, and $\boldsymbol{y}_i$ is a one-hot label vector.

Following the relationship between SVM [45] and LS-SVM [46], we minimize $\boldsymbol{\zeta}_i$ in (6) instead of $\boldsymbol{\zeta}_i^2$ in order to incorporate the LUPI paradigm easily. Therefore, we have

$$\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^{N}\boldsymbol{\zeta}_i$$
$$s.t. \quad \boldsymbol{h}(\boldsymbol{x}_i)\boldsymbol{w} = \boldsymbol{y}_i - \boldsymbol{\zeta}_i, \forall 1 \le i \le N. \quad (7)$$

Now we use the new learning paradigm to train the RVFL network. Given a set of additional privileged information $\{\tilde{\boldsymbol{x}}_i \in \mathbb{R}^d, i = 1, \ldots, N\}$ in the training process, the training data becomes $\{(\boldsymbol{x}_i, \tilde{\boldsymbol{x}}_i, \boldsymbol{y}_i)|\boldsymbol{x}_i \in \mathbb{R}^n, \tilde{\boldsymbol{x}}_i \in \mathbb{R}^d, \boldsymbol{y}_i \in \mathbb{R}^m, i = 1, \ldots, N\}$. In this new training set, $\boldsymbol{x}_i \in X$ is the original feature and in general the privileged feature $\tilde{\boldsymbol{x}}_i \in \tilde{X}$ belongs to the privileged feature space $\tilde{X}$ that is different from the original feature space $X$.

Following the formula of the SVM+ in [1], we can write the RVFL+ as

$$\min_{\boldsymbol{w}, \tilde{\boldsymbol{w}}} \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \frac{\gamma}{2}\|\tilde{\boldsymbol{w}}\|_2^2 + C\sum_{i=1}^{N}\zeta_i(\tilde{\boldsymbol{w}}, \tilde{\boldsymbol{h}}(\tilde{\boldsymbol{x}}_i))$$
$$s.t. \quad \boldsymbol{h}(\boldsymbol{x}_i)\boldsymbol{w} = \boldsymbol{y}_i - \zeta_i(\tilde{\boldsymbol{w}}, \tilde{\boldsymbol{h}}(\tilde{\boldsymbol{x}}_i)), \forall 1 \le i \le N. \quad (8)$$

where $\gamma$ is a regularization coefficient. Likewise to $\boldsymbol{h}(\boldsymbol{x}_i)$, $\tilde{\boldsymbol{h}}(\tilde{\boldsymbol{x}}_i)$ is also an enhanced layer output vector corresponding to the privileged feature $\tilde{\boldsymbol{x}}_i$, which can be calculated in the same fashion. $\zeta_i(\tilde{\boldsymbol{w}}, \tilde{\boldsymbol{h}}(\tilde{\boldsymbol{x}}_i))$ is the correcting function (or slack function) in the privileged feature space, and $\tilde{\boldsymbol{w}}$ is an output weight vector for the correcting function.

$$\zeta_i(\tilde{\boldsymbol{w}}, \tilde{\boldsymbol{h}}(\tilde{\boldsymbol{x}}_i)) = \tilde{\boldsymbol{h}}(\tilde{\boldsymbol{x}}_i)\tilde{\boldsymbol{w}} \quad (9)$$

Substituting (9) into (8), we have the primal form of the RVFL+ as follows.

$$\min_{\boldsymbol{w}, \tilde{\boldsymbol{w}}} \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \frac{\gamma}{2}\|\tilde{\boldsymbol{w}}\|_2^2 + C\sum_{i=1}^{N}\tilde{\boldsymbol{h}}(\tilde{\boldsymbol{x}}_i)\tilde{\boldsymbol{w}}$$
$$s.t. \quad \boldsymbol{h}(\boldsymbol{x}_i)\boldsymbol{w} = \boldsymbol{y}_i - \tilde{\boldsymbol{h}}(\tilde{\boldsymbol{x}}_i)\tilde{\boldsymbol{w}}, \forall 1 \le i \le N. \quad (10)$$

From (10), we note that the RVFL+ minimizes the objective function over both $\boldsymbol{w}$ and $\tilde{\boldsymbol{w}}$. Therefore, Not only the original features but also the privileged information determine meanwhile the separating hyperplane of the RVFL+ during the training stage.

Moreover, in contrasts to the primal form of the SVM+ in [1], the correcting function of the RVFL+ is either positive or negative. In other words, the RVFL+ does not consider a group of constraints $\zeta_i(\tilde{\boldsymbol{w}}, b, \phi(\tilde{\boldsymbol{x}_i})) \ge 0$ $(i = 1, \ldots, N)$. As a result, the number of constraints in the RVFL+ is at least $N$ less than that of the SVM+ for the binary classification, which results that the RVFL+ has much milder optimization constraints than the SVM+.

Furthermore, in order to address the optimization problem in (10), we construct the Lagrangian function $\mathscr{L}(\boldsymbol{w}, \tilde{\boldsymbol{w}}, \boldsymbol{\lambda})$ as

$$\min_{\boldsymbol{w}, \tilde{\boldsymbol{w}}, \boldsymbol{\lambda}} \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \frac{\gamma}{2}\|\tilde{\boldsymbol{w}}\|_2^2 + C\sum_{i=1}^{N}\tilde{\boldsymbol{h}}(\tilde{\boldsymbol{x}}_i)\tilde{\boldsymbol{w}}$$
$$- \sum_{i=1}^{N}\boldsymbol{\lambda}_i(\boldsymbol{h}(\boldsymbol{x}_i)\boldsymbol{w} - \boldsymbol{y}_i + \tilde{\boldsymbol{h}}(\tilde{\boldsymbol{x}}_i)\tilde{\boldsymbol{w}}) \quad (11)$$

where $\boldsymbol{\lambda} = [\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_N]^T$ are Lagrange multipliers.

To find solutions, we use the KKT condition to calculate the saddle points of the Lagrangian function $\mathscr{L}(\boldsymbol{w}, \tilde{\boldsymbol{w}}, \boldsymbol{\lambda})$ with respect to $\boldsymbol{w}$, $\tilde{\boldsymbol{w}}$ and $\boldsymbol{\lambda}$.

$$\frac{\partial\mathscr{L}(\boldsymbol{w}, \tilde{\boldsymbol{w}}, \boldsymbol{\lambda})}{\partial\boldsymbol{w}} = 0 \longrightarrow \boldsymbol{w} = \boldsymbol{H}^T\boldsymbol{\lambda} \quad (12)$$
$$\frac{\partial\mathscr{L}(\boldsymbol{w}, \tilde{\boldsymbol{w}}, \boldsymbol{\lambda})}{\partial\tilde{\boldsymbol{w}}} = 0 \longrightarrow \tilde{\boldsymbol{w}} = \frac{1}{\gamma}(\tilde{\boldsymbol{H}}^T\boldsymbol{\lambda} - \tilde{\boldsymbol{H}}^T C\mathbf{1}) \quad (13)$$
$$\frac{\partial\mathscr{L}(\boldsymbol{w}, \tilde{\boldsymbol{w}}, \boldsymbol{\lambda})}{\partial\boldsymbol{\lambda}_i} = 0 \longrightarrow \boldsymbol{h}(\boldsymbol{x}_i)\boldsymbol{w} - \boldsymbol{y}_i + \tilde{\boldsymbol{h}}(\tilde{\boldsymbol{x}}_i)\tilde{\boldsymbol{w}} = 0$$
$$\forall 1 \le i \le N. \quad (14)$$

where $\mathbf{1} \in \mathscr{R}^{N \times m}$ is an identify matrix. $\tilde{\boldsymbol{H}}$ is also a concatenated output matrix from the enhancement nodes, which corresponds to the privileged features.

Substituting (12) and (13) into (14), we have

$$\boldsymbol{H}\boldsymbol{H}^T\boldsymbol{\lambda} + \frac{1}{\gamma}\tilde{\boldsymbol{H}}\tilde{\boldsymbol{H}}^T(\boldsymbol{\lambda} - C\mathbf{1}) = \boldsymbol{Y} \quad (15)$$

We can further reformulate (15) as

$$(\boldsymbol{H}\boldsymbol{H}^T + \frac{1}{\gamma}\tilde{\boldsymbol{H}}\tilde{\boldsymbol{H}}^T)\boldsymbol{\lambda} = \boldsymbol{Y} - \frac{C\mathbf{1}}{\gamma}\tilde{\boldsymbol{H}}\tilde{\boldsymbol{H}}^T \quad (16)$$

Combining (12) and (16), we obtain the closed-form solution to the RVFL+ as follows.

$$\boldsymbol{w} = \boldsymbol{H}^T(\boldsymbol{H}\boldsymbol{H}^T + \frac{1}{\gamma}\tilde{\boldsymbol{H}}\tilde{\boldsymbol{H}}^T)^{-1}(\boldsymbol{Y} - \frac{C\mathbf{1}}{\gamma}\tilde{\boldsymbol{H}}\tilde{\boldsymbol{H}}^T) \quad (17)$$

According to the ridge regression [25], we also impose an additional term $\frac{\boldsymbol{I}}{C}$ in order to avoid singularity and guarantee the stability of the RVFL+. As a result, we can achieve the aftermost closed-form solution to the RVFL+ as

$$\boldsymbol{w} = \boldsymbol{H}^T(\boldsymbol{H}\boldsymbol{H}^T + \frac{1}{\gamma}\tilde{\boldsymbol{H}}\tilde{\boldsymbol{H}}^T + \frac{\boldsymbol{I}}{C})^{-1}(\boldsymbol{Y} - \frac{C\mathbf{1}}{\gamma}\tilde{\boldsymbol{H}}\tilde{\boldsymbol{H}}^T) \quad (18)$$

Consequently, the output function of the RVFL+ is defined as

$$f(\boldsymbol{x}) = \boldsymbol{h}(\boldsymbol{x})\boldsymbol{w} =$$
$$\boldsymbol{h}(\boldsymbol{x})\boldsymbol{H}^T(\boldsymbol{H}\boldsymbol{H}^T + \frac{1}{\gamma}\tilde{\boldsymbol{H}}\tilde{\boldsymbol{H}}^T + \frac{\boldsymbol{I}}{C})^{-1}(\boldsymbol{Y} - \frac{C\mathbf{1}}{\gamma}\tilde{\boldsymbol{H}}\tilde{\boldsymbol{H}}^T) \quad (19)$$

In addition, we can obtain straightforwardly the output function $f_{test}(\boldsymbol{z}) = \boldsymbol{h}(\boldsymbol{z})\boldsymbol{w}$ in the test stage, when using the test data $\boldsymbol{z}$ instead of the training data $\boldsymbol{x}$.

The pseudo-code of the RVFL+ is summarized in Algorithm 1.

For the SVM+ [1], the major challenge in terms of optimization arises from the constraint $\sum_{i=1}^{N}(\varphi_i + \psi_i - C) = 0$, where

**Algorithm 1** Random Vector Functional Link Networks with Privileged Information: RVFL+

---

**Input:** A set of training data $\{(\boldsymbol{x}_i, \tilde{\boldsymbol{x}}_i, \boldsymbol{y}_i) | \boldsymbol{x}_i \in \mathbb{R}^n, \tilde{\boldsymbol{x}}_i \in \mathbb{R}^d, \boldsymbol{y}_i \in \mathbb{R}^m, \forall 1 \le i \le N\}$; a nonlinear activation function $G(\cdot)$; the number of enhancement nodes $P$; the user-specified coefficients $C$, $\gamma$ and $u$.

**Output:** the output weight vector $\boldsymbol{w}$

1: Initialize randomly the weights $\boldsymbol{a}$ and biases $\boldsymbol{b}$ between the input layer and the enhancement nodes from a uniform distribution within $[-u, u]$ and $[0, u]$, respectively, and then, these generated weights and biases are fixed;
2: Calculate the output matrix $\boldsymbol{H}$ using (2);
3: Calculate the output matrix $\tilde{\boldsymbol{H}}$ in the same fashion;
4: Calculate the output weight vector $\boldsymbol{w}$ of RVFL+ using (18);
5: **return** the output weight vector $\boldsymbol{w}$

---

$\varphi_i$ and $\psi_i$ are Lagrange multipliers in the SVM+. Since these two sets of Lagrange multipliers need to be considered at the same time, which makes the optimization become difficult [5]. In addition to this above constraint, the RVFL+ also eliminates the other constraint $\sum_{i=1}^{N} \varphi_i y_i = 0$ in the SVM+. As a result, the RVFL+ has much simpler optimization constraints than the SVM+, and can achieve a closed-form solution.

Moreover, the RVFL+ is an unified learning algorithm for all binary, multiclass classification and regression, and the output function in (19) can be straightforwardly applied for all three cases.

- Binary Classification: The predicted label of the test sample is determined by

$$\hat{y} = sign(f_{test}(\boldsymbol{z})) \tag{20}$$

- Multiclass Classification: We adopt the one-vs.-all (OvA) strategy to determine the predicted label in the multiclass classification. Let $f_{test}^k(\boldsymbol{z})$ be the output function of the $k$-th output nodes. The predicted label of the test sample is determined by

$$\hat{y} = \arg \max_{k \in 1, \dots, m} f_{test}^k(\boldsymbol{z}) \tag{21}$$

- Regression: The predicted value is equal to the output function $f_{test}(\boldsymbol{z})$ of the RVFL+

$$\hat{y} = f_{test}(\boldsymbol{z}) \tag{22}$$

### C. Kernel Extension

In this section, we derive a kernel based random vector functional-link network with privileged information (KRVFL+ for short). There are two major advantages over the RVFL+. The KRVFL+ no longer considers the number of enhancement nodes, instead, the KRVFL+ maps the input data into a reproducing kernel Hilbert space (RKHS) in order to construct a Mercer kernel. On the other hand, the KRVFL+ is much more robust than the RVFL+. Since the KRVFL+ does not perform the random affine transformation between the input layer and enhancement nodes, and the enhanced layer output matrix is fixed when using kernel tricks. As a consequence, the

generalization performance of the KRVFL+ can be improved in terms of effectiveness and stability in most real-world tasks.

Likewise to [26], we reformulate (19) as

$$\boldsymbol{f}(\boldsymbol{x}) = [\boldsymbol{h}_1 \quad \boldsymbol{h}_2] \begin{bmatrix} \boldsymbol{H}_1^T \\ \boldsymbol{H}_2^T \end{bmatrix} \times$$

$$([\boldsymbol{H}_1 \quad \boldsymbol{H}_2] \begin{bmatrix} \boldsymbol{H}_1^T \\ \boldsymbol{H}_2^T \end{bmatrix} + \frac{1}{\gamma} [\tilde{\boldsymbol{H}}_1 \quad \tilde{\boldsymbol{H}}_2] \begin{bmatrix} \tilde{\boldsymbol{H}}_1^T \\ \tilde{\boldsymbol{H}}_2^T \end{bmatrix} + \frac{\boldsymbol{I}}{C})^{-1} \times$$

$$(\boldsymbol{Y} - \frac{C\boldsymbol{1}}{\gamma} [\tilde{\boldsymbol{H}}_1 \quad \tilde{\boldsymbol{H}}_2] \begin{bmatrix} \tilde{\boldsymbol{H}}_1^T \\ \tilde{\boldsymbol{H}}_2^T \end{bmatrix}) \tag{23}$$

We can further simplify (23) as

$$\boldsymbol{f}(\boldsymbol{x}) = (\boldsymbol{h}_1 \boldsymbol{H}_1^T + \boldsymbol{h}_2 \boldsymbol{H}_2^T) \times$$

$$(\boldsymbol{H}_1 \boldsymbol{H}_1^T + \boldsymbol{H}_2 \boldsymbol{H}_2^T + \tilde{\boldsymbol{H}}_1 \tilde{\boldsymbol{H}}_1^T + \tilde{\boldsymbol{H}}_2 \tilde{\boldsymbol{H}}_2^T + \frac{\boldsymbol{I}}{C})^{-1} \times$$

$$(\boldsymbol{Y} - \frac{C\boldsymbol{1}}{\gamma} (\tilde{\boldsymbol{H}}_1 \tilde{\boldsymbol{H}}_1^T + \tilde{\boldsymbol{H}}_2 \tilde{\boldsymbol{H}}_2^T)) \tag{24}$$

We define the kernel matrices for the KRVFL+ as

$$\boldsymbol{\Omega}_1 = \boldsymbol{H}_1 \boldsymbol{H}_1^T : \Omega_{1ij} = K_1(\boldsymbol{x}_i, \boldsymbol{x}_j)$$
$$\boldsymbol{\Omega}_2 = \boldsymbol{H}_2 \boldsymbol{H}_2^T : \Omega_{2ij} = K_2(\boldsymbol{x}_i, \boldsymbol{x}_j)$$
$$\tilde{\boldsymbol{\Omega}}_1 = \tilde{\boldsymbol{H}}_1 \tilde{\boldsymbol{H}}_1^T : \tilde{\Omega}_{1ij} = \tilde{K}_1(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j)$$
$$\tilde{\boldsymbol{\Omega}}_2 = \tilde{\boldsymbol{H}}_2 \tilde{\boldsymbol{H}}_2^T : \tilde{\Omega}_{2ij} = \tilde{K}_2(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j) \tag{25}$$

where $K_1$ and $\tilde{K}_1$ are linear kernels, as well as $K_2$ and $\tilde{K}_2$ are general Mercer kernels such as Gaussian kernel, polynomial kernel, and wavelet kernel, etc.

Substituting (25) into (24), we have

$$\boldsymbol{f}_{kernel}(\boldsymbol{x}) = \left( \begin{bmatrix} K_1(\boldsymbol{x}, \boldsymbol{x}_1) \\ \vdots \\ K_1(\boldsymbol{x}, \boldsymbol{x}_N) \end{bmatrix} + \begin{bmatrix} K_2(\boldsymbol{x}, \boldsymbol{x}_1) \\ \vdots \\ K_2(\boldsymbol{x}, \boldsymbol{x}_N) \end{bmatrix} \right) \times$$

$$(\boldsymbol{\Omega}_1 + \boldsymbol{\Omega}_2 + \frac{1}{\gamma}(\tilde{\boldsymbol{\Omega}}_1 + \tilde{\boldsymbol{\Omega}}_2) + \frac{\boldsymbol{I}}{C})^{-1}(\boldsymbol{Y} - \frac{C\boldsymbol{1}}{\gamma}(\tilde{\boldsymbol{\Omega}}_1 + \tilde{\boldsymbol{\Omega}}_2)) \tag{26}$$

where the output weight vector $\boldsymbol{w}_{kernel}$ of the KRVFL+ is defined as

$$\boldsymbol{w}_{kernel} = (\boldsymbol{\Omega}_1 + \boldsymbol{\Omega}_2 + \frac{1}{\gamma}(\tilde{\boldsymbol{\Omega}}_1 + \tilde{\boldsymbol{\Omega}}_2) + \frac{\boldsymbol{I}}{C})^{-1}$$
$$\times (\boldsymbol{Y} - \frac{C\boldsymbol{1}}{\gamma}(\tilde{\boldsymbol{\Omega}}_1 + \tilde{\boldsymbol{\Omega}}_2)) \tag{27}$$

According to the property of the Mercer kernel [25], we can achieve the aftermost formula of the KRVFL+ as follows.

$$\boldsymbol{f}_{kernel}(\boldsymbol{x}) = \left( \begin{bmatrix} K_1(\boldsymbol{x}, \boldsymbol{x}_1) \\ \vdots \\ K_1(\boldsymbol{x}, \boldsymbol{x}_N) \end{bmatrix} + \begin{bmatrix} K_2(\boldsymbol{x}, \boldsymbol{x}_1) \\ \vdots \\ K_2(\boldsymbol{x}, \boldsymbol{x}_N) \end{bmatrix} \right) \times$$

$$(\boldsymbol{\Omega} + \frac{1}{\gamma}\tilde{\boldsymbol{\Omega}} + \frac{\boldsymbol{I}}{C})^{-1}(\boldsymbol{Y} - \frac{C\boldsymbol{1}}{\gamma}\tilde{\boldsymbol{\Omega}}) \tag{28}$$

where $\boldsymbol{\Omega} = \boldsymbol{\Omega}_1 + \boldsymbol{\Omega}_2$ and $\tilde{\boldsymbol{\Omega}} = \tilde{\boldsymbol{\Omega}}_1 + \tilde{\boldsymbol{\Omega}}_2$.

Likewise to the RVFL+, the KRVFL+ can straightforwardly calculate the output function $f_{kernel-test}(z)$ in the test stage, which is shown as

$$f_{kernel-test}(z) = \left( \begin{bmatrix} K_1(z, x_1) \\ \vdots \\ K_1(z, x_N) \end{bmatrix} + \begin{bmatrix} K_2(z, x_1) \\ \vdots \\ K_2(z, x_N) \end{bmatrix} \right) \times$$
$$(\Omega + \frac{1}{\gamma}\tilde{\Omega} + \frac{I}{C})^{-1}(Y - \frac{C\mathbf{1}}{\gamma}\tilde{\Omega}) \tag{29}$$

The pseudo-code of the KRVFL+ is summarized in Algorithm 2.

---

**Algorithm 2** Kernel based Random Vector Functional Link Networks with Privileged Information: KRVFL+

---

**Input:** A set of training data $\{(x_i, \tilde{x}_i, y_i)|x_i \in \mathbb{R}^n, \tilde{x}_i \in \mathbb{R}^d, y_i \in \mathbb{R}^m, \forall 1 \le i \le N\}$; a Mercer kernel function (for example Gaussian kernel); the user-specified coefficients $C$ and $\gamma$.

**Output:** the output weight vector $w_{kernel}$.

1: Calculate the linear kernel functions $\Omega_1$ and $\tilde{\Omega}_1$ corresponding to $x$ and $\tilde{x}$, respectively;
2: Calculate the general Mercer kernel functions (for example Gaussian kernel) $\Omega_2$ and $\tilde{\Omega}_2$ corresponding to $x$ and $\tilde{x}$, respectively;
3: Calculate the output weight vector $w_{kernel}$ of KRVFL+ using (27);
4: **return** the output weight vector $w_{kernel}$.

---

## IV. THEORETICAL ANALYSIS OF RVFL+

In this section, we investigate the statistical property of the RVFL+, and only consider the binary classification for simplicity. Following the Rademacher complexity, we provide a tight generalization error bound for the RVFL+. We assume all sets considered in this paper are measurable. First of all, we give the following fact, which is a necessary condition for the Rademacher complexity.

**Proposition 1.** The loss function $\ell$ in the RVFL+ satisfies Lipschitz continuity, and there exists a positive Lipschitz constant $K$, that is, for $\forall x, y \in R^n$

$$\|\ell(x) - \ell(y)\| \le K\|x - y\| \tag{30}$$

where $\| \cdot \|$ is a norm function.

*Proof.* It is straightforward that the loss function $\ell$ in the RVFL+ is a norm function, which must satisfy Lipschitz continuity. $\square$

The novel generalization error bound is dependent on the Rademacher complexity, and thus, we give the definition of the Rademacher complexity.

**Definition 2.** [24] Given a set of i.i.d. samples $\mu_1, \ldots, \mu_M$, where $\mu_i \in U, \forall 1 \le i \le M$. Let $\mathscr{F}$ be a family of functions mapping from the space $U$ to an output space. Then, we define the Rademacher complexity of $\mathscr{F}$ as

$$\mathscr{R}_M(\mathscr{F}) = \mathbb{E}\left[ \sup_{f \in \mathscr{F}} \frac{1}{M} \sum_{i=1}^{M} f(\mu_i)\epsilon_i \right] \tag{31}$$

where $\epsilon_i, i = 1, \ldots, M$ are i.i.d $\{\pm 1\}$-valued Bernoulli random variables drawn at equal probability and are independent of inputs.

According to [24], we illustrate the general mathematical formula of the generalization error bound based on the Rademacher complexity in the following theorem.

**Theorem 3.** [24] Let $\mathfrak{L}(f)$ and $\hat{\mathfrak{L}}(f)$ be the generalization error bound and the empirical error bound, respectively, and let $\mathscr{F}$ be a family of functions. For a Lipschitz continuous loss function $\ell$ bounded by $c$, with probability at least $1 - \delta$ ($\delta \in (0, 1)$) over the samples with $M$, for all $f \in \mathscr{F}$, we have

$$\mathfrak{L}(f) \le \hat{\mathfrak{L}}(f) + 2K\mathscr{R}_M(\mathscr{F}) + c\sqrt{\frac{ln(1/\delta)}{2M}} \tag{32}$$

From Theorem 3, we note that the generalization error is bounded by the Rademacher complexity. Therefore, according to [47], we give a Rademacher complexity of the proposed RVFL+, which serves to bound the generalization error of the RVFL+. The following Lemma is helpful in bounding the Rademacher complexity.

**Lemma 4.** A function $F(\cdot) = \| \cdot \|_2^2$ is a $\sigma$-strongly convex ($\sigma \in (0, 2]$) with respect to itself, that is, for $\forall \alpha \in [0, 1]$ and $u, v \in \mathbb{R}^d$, we have

$$F(\alpha u + (1 - \alpha)v) \le \alpha F(u) + (1 - \alpha)F(v)$$
$$- \frac{\sigma}{2}\alpha(1 - \alpha)\|u - v\|_2^2 \tag{33}$$

*Proof.*

$$F(\alpha u + (1 - \alpha)v) - \alpha F(u) - (1 - \alpha)F(v)$$
$$= \|\alpha u + (1 - \alpha)v\|_2^2 - \alpha\|u\|_2^2 - (1 - \alpha)\|v\|_2^2$$
$$= -\alpha(1 - \alpha)\|u\|_2^2 + 2\alpha(1 - \alpha)\langle u, v \rangle - \alpha(1 - \alpha)\|v\|_2^2$$
$$= -\alpha(1 - \alpha)(\|u\|_2^2 - 2\langle u, v \rangle + \|v\|_2^2)$$
$$= -\alpha(1 - \alpha)\|u - v\|_2^2 \le -\frac{\sigma}{2}\alpha(1 - \alpha)\|u - v\|_2^2 \tag{34}$$

The inequality follows by an obvious inequality $-\alpha(1 - \alpha)\|u - v\|_2^2 \le 0$. When $\sigma = 2$, the equality in this Lemma holds. $\square$

The Rademacher complexity is one measure of the number of a family $\mathscr{F}$ of functions. Now we use convex duality to show that the Rademacher complexity is bounded by the number of inputs as follows.

**Theorem 5.** Define $\mathscr{H} = \{h : \|h\|_2 \le Z\}$ and $\mathscr{B} = \{w \in S : \|w\|_2^2 \le B^2\}$, where $S \in \mathbb{R}^M$ is a subset of the dual space to the enhanced layer output vector space. The $\| \cdot \|_2^2$ is 2-strongly convex on $\mathbb{R}^M$ with respect to itself. Then, the Rademacher complexity is bounded by

$$\mathscr{R}_M(\mathscr{F}) \le ZB\sqrt{\frac{1}{M}} \tag{35}$$

*Proof.* Given a set of $\{\boldsymbol{h}_i\}$ $(i = 1, \ldots, M)$ such that $\|\boldsymbol{h}_i\|_2 \leq Z$. Define $\boldsymbol{\theta} = \frac{1}{M}\sum_i \epsilon_i \boldsymbol{h}_i$, where $\epsilon_i, i = 1, \ldots, M$ are Rademacher random variables defined above such that $\mathbb{E}[\epsilon_i] = 0$ and $\mathbb{E}[\epsilon_i^2] = 1$.

Using Fenchel-Young inequality, we have

$$\langle \boldsymbol{w}, \kappa\boldsymbol{\theta} \rangle \leq F(\boldsymbol{w}) + F^*(\boldsymbol{w}) \tag{36}$$

where $\kappa$ is an arbitrary positive real, and $F^*(\cdot)$ is the Fenchel conjugate of $F(\cdot)$.

Since the dual norm of the $l_2$ norm is itself, so we further have

$$\langle \boldsymbol{w}, \boldsymbol{\theta} \rangle \leq \frac{\|\boldsymbol{w}\|_2^2}{\kappa} + \frac{\|\kappa\boldsymbol{\theta}\|_2^2}{\kappa} \tag{37}$$

Following the condition of this theorem, we have

$$\sup_{\boldsymbol{w} \in \mathscr{B}} \langle \boldsymbol{w}, \boldsymbol{\theta} \rangle \leq \frac{B^2}{\kappa} + \frac{\|\kappa\boldsymbol{\theta}\|_2^2}{\kappa} \tag{38}$$

We calculate expectation over $\epsilon_i$ on both sides and obtain

$$\mathbb{E}\big[\sup_{\boldsymbol{w} \in \mathscr{B}} \langle \boldsymbol{w}, \boldsymbol{\theta} \rangle\big] \leq \frac{B^2}{\kappa} + \frac{\mathbb{E}[\|\kappa\boldsymbol{\theta}\|_2^2]}{\kappa} \tag{39}$$

We have proven $\|\cdot\|_2^2$ is a $\sigma$-strongly convex in Lemma 4. Therefore, we can use the result of Lemma 2 in [47]. As a result, we can achieve the upper bound of $\mathbb{E}[\|\kappa\boldsymbol{\theta}\|_2^2]$ as

$$\mathbb{E}[\|\kappa\boldsymbol{\theta}\|_2^2] \leq \frac{\kappa^2 Z^2}{4M} \tag{40}$$

Substituting (40) into (39), we have

$$\mathbb{E}\big[\sup_{\boldsymbol{w} \in \mathscr{B}} \langle \boldsymbol{w}, \boldsymbol{\theta} \rangle\big] \leq \frac{B^2}{\kappa} + \frac{\kappa Z^2}{4M} \tag{41}$$

We set $\kappa$ as $\sqrt{\frac{4MB^2}{Z^2}}$, and substitute it into (41)

$$\mathbb{E}\big[\sup_{\boldsymbol{w} \in \mathscr{B}} \langle \boldsymbol{w}, \boldsymbol{\theta} \rangle\big] \leq ZB\sqrt{\frac{1}{M}} \tag{42}$$

which is the desired result. $\square$

Now we can bound the generalization error using the proved Rademacher complexity as follows.

**Theorem 6.** For the Lipschitz continuous loss function $\ell$ with Lipschitz constant $K$ in the RVFL+, with probability at least $1-\delta$ $(\delta \in (0, 1))$ over the inputs with length $M$, for all $f \in \mathscr{F}$, the generalization risk minimization of the RVFL+ is bounded by

$$\mathfrak{L}(f) \leq \hat{\mathfrak{L}}(f) + 2KZB\sqrt{\frac{1}{M}} + KZB\sqrt{\frac{ln(1/\delta)}{2M}} \tag{43}$$

*Proof.* We combine Theorem 3 and 5, and set the bounded constant $c = KZB$. Then, we can obtain the desired result. $\square$

## V. EXPERIMENTS AND RESULTS

In this section, we conduct several experiments to evaluate the proposed RVFL+ and KRVFL+ on 14 real-world datasets, including 1 binary classification dataset, 8 multi-class classification datasets and 5 regression datasets. Ten trials for each algorithm are carried out, and the average results are reported. All simulations are carried out in a Matlab 2015b environment running in a PC machine with an Inter(R) Core(TM) i7-6700HD 2.60 GHZ CPU and 32 GB of RAM.

### A. Evaluation on MNIST+

*1) Dataset:* The MNIST+ dataset is a popular testbed used in [1], [5], [6] for verifying the performance of the algorithms based on the LUPI. The MNIST+ is a handwritten digit recognition dataset, consisting of the images of the digits '5' and '8'. There are 100 training samples, 4,000 validation samples and 1,866 test samples in MNIST+. The original 28-by-28 gray-scale images from MNIST [48] are resized into 10-by-10 gray-scale images in order to increase the challenge. In the MNIST+, each sample in the training set and test set contains 100-dimensional attributes used as the main features. Moreover, the additional privileged information in the MNIST+ is 21-dimensional texture features based data, which is a holistic description of each image. We refer interested readers to [1] for details.

*2) Parameters selection:* For the RVFL+, we determine empirically the hyper-parameters $C$, $\gamma$, $u$ and $L$, as well as the nonlinear activation function on the validation set. First of all, we select the nonlinear activation function. The Table I reports the performance of the RVFL+ with different activation functions in terms of accuracy. As shown by Table I, the RVFL+ with the triangular basis function outperforms others, and thus, we use the triangular basis function as the activation function in this case. However, we found that there does not exist a general rule for choosing the activation function. Therefore, we need to determine the activation function in the RVFL+ for different tasks.

TABLE I: The evaluation of the RVFL+ with different activation functions

| Algorithm | Activation Function | Accuracy (%) |
|---|---|---|
| RVFL+ | Sigmoid function | $85.22 \pm 0.38$ |
| | Sine function | $85.06 \pm 1.06$ |
| | Hardlim function | $87.01 \pm 1.56$ |
| | Triangular basis function | $\mathbf{90.41 \pm 0.74}$ |
| | Radial basis function | $84.36 \pm 0.54$ |

The user-defined parameters $C$ and $\gamma$ are chosen through a random search [49] within $[10^{-5}, 10^5]$, and the $u$ is also selected from the interval $[2^{-5}, 2^{-4.5}, \ldots, 2^{4.5}, 2^5]$ in the same fashion. In the Figure 2, We only show parts of results of selecting the parameters $C$ and $\gamma$, and ignore numerous ones having the worse performance. As shown in the Figure 2, when $C$ and $\gamma$ are set as 1 and 1,000 respectively, the RVFL+ can achieve the best performance. Therefore, we determine $C$ and $\gamma$ as 1 and 1,000 in the following comparison. Additionally, when the number of enhancement nodes $P$ is sufficient large,

the RVFL+ can achieve good generalization performance. Therefore balancing the effectiveness and efficiency, $P$ is determined as 1,000 in all experiments. Furthermore, we found that a 'suitable' $u$ can improve around 2% accuracy for the RVFL+, and thus, we empirically determine an optimal $u$ in different tasks. In this experiment, the positive factor $u$ is empirically set as $2^{2.5}$. Due to space limitations, we omit these procedures of selecting $P$ and $u$.
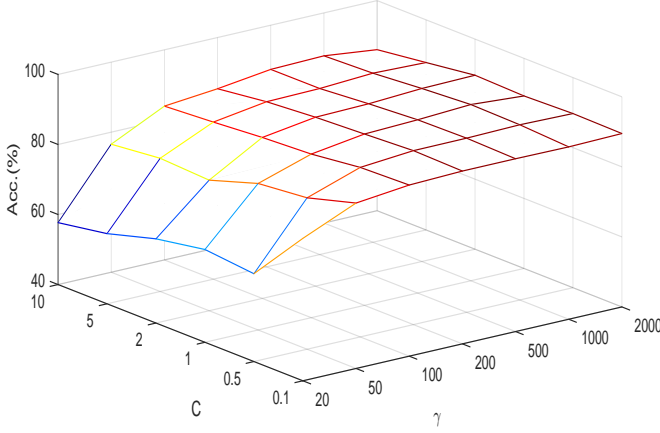


Fig. 2: The performance of RVFL+ with different user-defined parameters $C$ and $\gamma$, in which the RVFL+ uses the triangular basis function as the activation function.

There are two kernel functions in the KRVFL+. In all simulations, we use the commonly-used Gaussian kernel function as the general Mercer kernel, while the other kernel function is always a linear kernel function. The user-specified kernel parameter $\tau$ as well as $C$ and $\gamma$ are determined on the validation set. Seen from the Figure 3a, when the $\gamma$ is sufficient large (more than 5,000), the KRVFL+ is insensitive to this parameter. Therefore, we set $\gamma$ as 5,000 for KRVFL+ in all experiments. The Figure 3b illustrates the performance of KRVFL+ with different parameters $C$ and $\tau$, when $\gamma = 5,000$. We found that when $C$ and $\tau$ are in the interval $[1, 2]$ and $[0.02, 0.03]$ respectively, the RVFL+ can achieve the best performance in this case.

For other approaches, due to space limitations, we omit the procedure of selecting the hyper-parameters. We empirically determine these user-specified parameters on the validation set, and pick the one having the best performance.

*3) Experimental Results and Discussion:* We compare with state-of-the-art approaches on MNIST+ in terms of accuracy (%) and time (s), including SVM [50], gSMO-SVM+ [5], CVX-SVM+ [51], MAT-SVM+ [6], Fast SVM+ [6] and RVFL [19]. The experimental results are shown in Table II.

From Table II, we see that the KRVFL+ has the best performance in terms of effectiveness and efficiency. Although the proposed RVFL+ is slightly worse than the kernel-based SVM+, this algorithm is much better than all linear SVM+. In contrast to SVM and RVFL, this experimental results illustrates that benefits of the LUPI-based approaches accrue with little additional computational cost.

TABLE II: The comparisons with state-of-the-art approaches on MNIST+ in terms of accuracy (%) and time (s)

| Algorithms | Kernel/Activation Function | Acc. (%) | time (s) |
|---|---|---|---|
| SVM [50] | Linear kernel | 81.73 | 0.009 |
| gSMO-SVM+ [5] | Linear kernel | 84.35 | 1.103 |
| Fast SVM+ [6] | Linear kernel | 84.62 | 0.146 |
| SVM [50] | Gaussian kernel | 92.34 | 0.009 |
| gSMO-SVM+ [5] | Gaussian kernel | 92.77 | 1.206 |
| CVX-SVM+ [51] | Gaussian kernel | 93.14 | 11.746 |
| MAT-SVM+ [6] | Gaussian kernel | 93.14 | 0.572 |
| Fast SVM+ [6] | Gaussian kernel | 93.15 | 0.039 |
| RVFL [19] | Triangular basis | 87.08 | 0.021 |
| RVFL+ | Triangular basis | 90.40 | 0.027 |
| KRVFL+ | Gaussian kernel | **93.71** | 0.005 |

### B. Evaluation on UCI Classification Datasets

In this section, we compare with Fast SVM+ [6], MAT-SVM+ [6], SVM [50] and RVFL [19] on 8 real-world multi-class classification datasets from UCI machine learning repository [52], which cover a large range of multi-class classification tasks. The previous work [6] has verified that the Fast SVM+ outperforms other state-of-the-art LUPI-based algorithms. To avoid duplication, these algorithms are not included for comparison. The statistics of the UCI classification datasets are illustrated in Table III, including the number of training and test data, attributes and classes. We add white noise with 10 dBW to both training and test samples. While the original training data without white noise is used as privileged information for all LUPI-based algorithms. We use $L_1$ normalization to pre-process all samples. We use two-fold cross validation (CV) for Shuttle, five-fold CV for Abalone/Red Wine Quality/White Wine Quality, and ten-fold CV for the rest of the datasets. We found that both RVFL+ and RVFL can achieve the best performance, when using the sigmoid function as the nonlinear activation function in this experiment. Due to space limitations, we only report the final experimental results of all comparisons, and omit the procedure of selecting hyper-parameters and pick the one having the best performance. The experimental results are reported in Table IV.

TABLE III: The statistics of UCI classification datasets

| Problems | Training Data | Test Data | Attributes | Classes |
|---|---|---|---|---|
| Iris | 90 | 60 | 4 | 3 |
| Glass | 140 | 74 | 9 | 5 |
| Abalone | 2,000 | 2,177 | 8 | 3 |
| Wine | 100 | 78 | 13 | 3 |
| Red Wine Quality | 1,000 | 599 | 11 | 3 |
| White Wine Quality | 3,000 | 1,898 | 11 | 7 |
| Shuttle | 43,500 | 14,500 | 9 | 7 |
| Segment | 100 | 110 | 19 | 7 |

Seen from Table IV, except for the white wine quality dataset and the red wine quality dataset, the KRVFL+ can
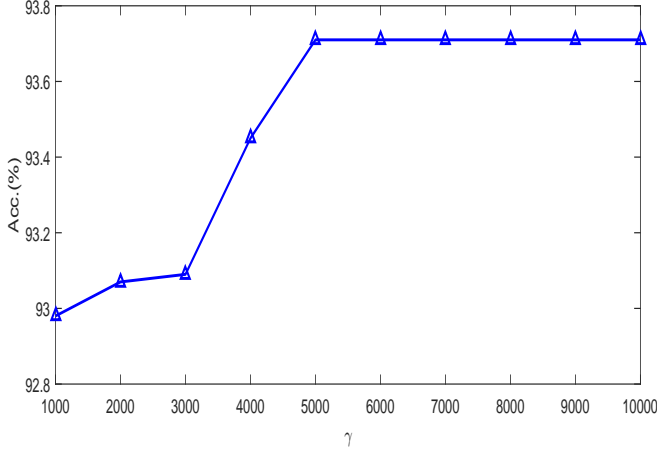
(a) The best performance of KRVFL+ corresponding to different $\gamma$.
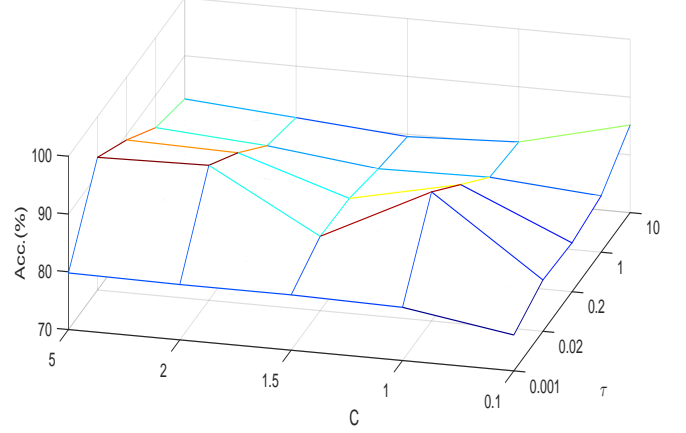
(b) The performance of KRVFL+ with different user-defined parameters $C$ and $\tau$, when $\gamma = 5,000$.

Fig. 3: Parameter selection for KRVFL+.

TABLE IV: The comparisons with state-of-the-art approaches on UCI classification datasets in terms of accuracy (%) and time (s)

| Problems | KRVFL+ | | RVFL+ | | RVFL [19] | | Fast SVM+ [6] | | MAT-SVM+ [6] | | SVM [50] | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Acc.(%) | time (s) | Acc.(%) | time (s) | Acc.(%) | time (s) | Acc.(%) | time (s) | Acc.(%) | time (s) | Acc.(%) | time (s) |
| Iris | **51.67** | 0.002 | 46.67 | 0.093 | 43.33 | 0.078 | 49.63 | 0.007 | 49.62 | 1.102 | 43.57 | 0.002 |
| Glass | **39.19** | 0.002 | 36.49 | 0.154 | 29.03 | 0.141 | 38.95 | 0.01 | 38.95 | 0.231 | 29.75 | 0.002 |
| Abalone | **43.50** | 0.219 | 42.77 | 0.643 | 42.31 | 0.578 | 42.90 | 1.194 | 42.87 | 37.13 | 42.09 | 0.134 |
| Wine | **83.00** | 0.003 | 74.36 | 0.259 | 61.54 | 0.234 | 82.05 | 0.016 | 81.98 | 0.250 | 61.77 | 0.004 |
| Red Wine Quality | 53.92 | 0.672 | 52.59 | 0.392 | 52.09 | 0.343 | **53.98** | 0.307 | 53.91 | 378.3 | 52.14 | 0.082 |
| White Wine Quality | 47.74 | 0.514 | **49.00** | 0.824 | 45.47 | 0.719 | 47.13 | 3.592 | 47.02 | > 3600 | 45.86 | 1.034 |
| Shuttle | **98.48** | 1087.6 | 97.18 | 5.982 | 96.56 | 5.375 | 98.33 | 976.4 | 98.21 | > 3600 | 96.62 | 713.2 |
| Segment | **79.09** | 0.743 | 77.27 | 1.792 | 73.64 | 1.676 | 78.52 | 0.008 | 78.41 | 0.491 | 73.38 | 0.003 |

achieve the best performance in terms of accuracy and learning speed among all comparisons. While the RVFL+ outperforms other comparisons including KRVFL+ on the white wine quality dataset. Additionally, likewise to the above experiment, the performance of the algorithms with the LUPI paradigm including KRVFL+, RVFL+, Fast SVM+ and MAT-SVM+ are better remarkably than both SVM and RVFL.

### C. Evaluation on Regression Datasets

In this section, we compare with SVR [50] and RVFL [19] on 5 real-world regression datasets, which cover different categories from [53]. The statistics of these regression datasets are summarized in Table V. All samples are also added white noise with 10dBW, and the original training data without white noise is used as privileged information. We use $L_1$ normalization to pre-process all data. We use two-fold cross validation (CV) for SCM1D/SCM20D, ten-fold CV for the rest of the datasets. We also found that both RVFL+ and RVFL can achieve the best performance, when using the sigmoid function as the nonlinear activation function in this experiment. We also omit the procedure of selecting user-specified parameters.

The performance is measured by the commonly-used Root Mean Square Error (RMSE). The experimental results are shown in Table VI. The experimental results illustrate the advantage of the new learning paradigm. Although the RVFL+ and KRVFL+ achieve sightly better performance than the other two on EDM and SLUMP due to the small size of these two datasets, the two proposed algorithms have distinct advantage on the big size of datasets such as SCM1D and SCM20D.

TABLE V: The statistics of regression datasets

| Problems | Training Data | Test Data | Attributes |
|---|---|---|---|
| ANDRO | 23 | 26 | 30 |
| EDM | 140 | 74 | 16 |
| SLUMP | 53 | 50 | 7 |
| SCM1D | 5,000 | 4,803 | 280 |
| SCM20D | 6,000 | 2,966 | 61 |

### VI. CONCLUSION

In this paper, we first present a LUPI based random vector functional link network called RVFL+, which combines

TABLE VI: The comparisons with state-of-the-art approaches on regression datasets in terms of rmse and time (s)

| Problems | KRVFL+ | | RVFL+ | | RVFL [19] | | SVR [50] | |
|---|---|---|---|---|---|---|---|---|
| | RMSE | time (s) | RMSE | time (s) | RMSE | time (s) | RMSE | time (s) |
| ANDRO | **2.097** | 0.0004 | 2.151 | 0.1125 | 2.891 | 0.1094 | 2.853 | 0.0509 |
| EDM | 0.178 | 0.0014 | **0.175** | 0.0836 | 0.182 | 0.0781 | 0.179 | 0.0012 |
| SLUMP | 3.503 | 0.0005 | **3.501** | 0.1091 | 3.508 | 0.0781 | 3.512 | 0.0012 |
| SCM1D | **45.816** | 5.1495 | 49.007 | 22.328 | 51.618 | 23.755 | 51.962 | 12.723 |
| SCM20D | **99.47** | 7.958 | 103.08 | 5.347 | 109.91 | 4.9844 | 107.86 | 5.377 |

neural networks with the new LUPI paradigm to leverage the benefits of both. More significantly, the RVFL+ offers a new learning paradigm for the RVFL network, in which the additional privileged information is used to 'teach' the network during the training stage. Therefore, we note that the working mechanism of the newly-derived RVFL+ is in analogy to *Teacher-Student Interaction* [3] in human learning process. As a result, the RVFL+ can achieve better generalization performance than the RVFL in real-world tasks. In addition, from an optimization perspective, the RVFL+ has less and simpler optimization constraints than the SVM+ [1], which results that the RVFL+ can obtain a closed-form solution. Moreover, the RVFL+ can perform in conjunction with the kernel trick, which is defined as KRVFL+. The novel KRVFL+ is powerful enough to handle the more complicated nonlinear relationships between high-dimensional input data. Furthermore, we explore the statistical property of the proposed RVFL+, and establish a tight generalization error bound based on the Rademacher complexity for the RVFL+. Competitive experimental results on 14 diverse real-world datasets confirm the efficiency and effectiveness of the new RVFL+ and KRVFL+, which can achieve better generalization performance than the state-of-the-art algorithms.

## REFERENCES

[1] V. Vapnik and A. Vashist, "A new learning paradigm: Learning using privileged information," *Neural networks*, vol. 22, no. 5, pp. 544–557, 2009.

[2] D. Pechyony and V. Vapnik, "On the theory of learnining with privileged information," in *Advances in neural information processing systems*, 2010, pp. 1894–1902.

[3] V. Vapnik and R. Izmailov, "Learning using privileged information: similarity control and knowledge transfer." *Journal of machine learning research*, vol. 16, no. 20232049, p. 55, 2015.

[4] M. Grant, S. Boyd, and Y. Ye, "Cvx: Matlab software for disciplined convex programming," 2008.

[5] D. Pechyony, R. Izmailov, A. Vashist, and V. Vapnik, "Smo-style algorithms for learning using privileged information." in *DMIN*, 2010, pp. 235–241.

[6] W. Li, D. Dai, M. Tan, D. Xu, and L. Van Gool, "Fast algorithms for linear and kernel svm+," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2258–2266.

[7] D. Pechyony and V. Vapnik, "Fast optimization algorithms for solving svm+," *Stat. Learning and Data Science*, vol. 1, 2011.

[8] J. Feyereisl, S. Kwak, J. Son, and B. Han, "Object localization based on structural svm using privileged information," in *Advances in Neural Information Processing Systems*, 2014, pp. 208–216.

[9] S. Fouad, P. Tino, S. Raychaudhury, and P. Schneider, "Incorporating privileged information through metric learning," *IEEE transactions on neural networks and learning systems*, vol. 24, no. 7, pp. 1086–1098, 2013.

[10] X. Xu, W. Li, and D. Xu, "Distance metric learning using privileged information for face verification and person re-identification," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 12, pp. 3150–3162, 2015.

[11] M. Lapin, M. Hein, and B. Schiele, "Learning using privileged information: Svm+ and weighted svm," *Neural Networks*, vol. 53, pp. 95–108, 2014.

[12] V. Sharmanska, N. Quadrianto, and C. H. Lampert, "Learning to rank using privileged information," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 825–832.

[13] Y. Bengio, L. Yao, G. Alain, and P. Vincent, "Generalized denoising auto-encoders as generative models," in *Advances in Neural Information Processing Systems*, 2013, pp. 899–907.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[15] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[17] Y.-H. Pao and S. M. Phillips, "The functional link net and learning optimal control," *Neurocomputing*, vol. 9, no. 2, pp. 149–164, 1995.

[18] Y.-H. Pao and Y. Takefuji, "Functional-link net computing: theory, system architecture, and functionalities," *Computer*, vol. 25, no. 5, pp. 76–79, 1992.

[19] Y.-H. Pao, G.-H. Park, and D. J. Sobajic, "Learning and generalization characteristics of the random vector functional-link net," *Neurocomputing*, vol. 6, no. 2, pp. 163–180, 1994.

[20] B. Igelnik and Y.-H. Pao, "Stochastic choice of basis functions in adaptive function approximation and the functional-link net," *IEEE Transactions on Neural Networks*, vol. 6, no. 6, pp. 1320–1329, 1995.

[21] L. Zhang and P. N. Suganthan, "A comprehensive evaluation of random vector functional link networks," *Information sciences*, vol. 367, pp. 1094–1105, 2016.

[22] V. Vovk, "Kernel ridge regression," in *Empirical inference*. Springer, 2013, pp. 105–116.

[23] C. Saunders, A. Gammerman, and V. Vovk, "Ridge regression learning algorithm in dual variables." in *ICML*, 1998, pp. 515–521.

[24] P. L. Bartlett and S. Mendelson, "Rademacher and gaussian complexities: Risk bounds and structural results," *Journal of Machine Learning Research*, vol. 3, no. Nov, pp. 463–482, 2002.

[25] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[26] K.-K. Xu, H.-X. Li, and H.-D. Yang, "Kernel-based random vector functional-link network for fast learning of spatiotemporal dynamic processes," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2017.

[27] L. Liu and P. Fieguth, "Texture classification from random features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 574–586, 2012.

[28] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[29] ——, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[30] D. Wang and M. Li, "Stochastic configuration networks: Fundamentals and algorithms," *arXiv preprint arXiv:1702.03180*, 2017.

[31] ——, "Robust stochastic configuration networks with kernel density estimation for uncertain data regression," *Information Sciences*, 2017.

[32] L. Zhang and P. N. Suganthan, "A survey of randomized algorithms for training neural networks," *Information Sciences*, vol. 364, pp. 146–155, 2016.

[33] S. Scardapane and D. Wang, "Randomness in neural networks: an overview," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 7, no. 2, 2017.

[34] C. P. Chen and J. Z. Wan, "A rapid learning and dynamic stepwise updating algorithm for flat neural networks and the application to time-series prediction," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 29, no. 1, pp. 62–72, 1999.

[35] C. P. Chen, "A rapid supervised learning neural network for function interpolation and approximation," *IEEE Transactions on Neural Networks*, vol. 7, no. 5, pp. 1220–1230, 1996.

[36] J. C. Patra, R. N. Pal, B. Chatterji, and G. Panda, "Identification of nonlinear dynamic systems using functional link artificial neural networks," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 29, no. 2, pp. 254–262, 1999.

[37] S. Scardapane, D. Wang, M. Panella, and A. Uncini, "Distributed learning for random vector functional-link networks," *Information Sciences*, vol. 301, pp. 271–284, 2015.

[38] S. Scardapane, D. Wang, and A. Uncini, "Bayesian random vector functional-link networks for robust data modeling," *IEEE Transactions on Cybernetics*, 2017.

[39] Y. Ren, P. N. Suganthan, N. Srikanth, and G. Amaratunga, "Random vector functional link network for short-term electricity load demand forecasting," *Information Sciences*, vol. 367, pp. 1078–1093, 2016.

[40] C. P. Chen and Z. Liu, "Broad learning system: An effective and efficient incremental learning system without the need for deep architecture," *IEEE Transactions on Neural Networks and Learning Systems*, 2017.

[41] C. P. Chen and C.-Y. Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on big data," *Information Sciences*, vol. 275, pp. 314–347, 2014.

[42] R. Giryes, G. Sapiro, and A. M. Bronstein, "Deep neural networks with random gaussian weights: a universal classification strategy?" *IEEE Trans. Signal Processing*, vol. 64, no. 13, pp. 3444–3457, 2016.

[43] L. Wolf and A. Shashua, "Learning over sets using kernel principal angles," *Journal of Machine Learning Research*, vol. 4, no. Oct, pp. 913–931, 2003.

[44] P. L. Bartlett, "The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network," *IEEE transactions on Information Theory*, vol. 44, no. 2, pp. 525–536, 1998.

[45] C. Cortes and V. Vapnik, "Support vector machine," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[46] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters*, vol. 9, no. 3, pp. 293–300, 1999.

[47] S. M. Kakade, K. Sridharan, and A. Tewari, "On the complexity of linear prediction: Risk bounds, margin bounds, and regularization," in *Advances in neural information processing systems*, 2009, pp. 793–800.

[48] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[49] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.

[50] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[51] L. Liang and V. Cherkassky, "Connection between svm+ and multi-task learning," in *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*. IEEE, 2008, pp. 2048–2054.

[52] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[53] E. Spyromitros-Xioufis, G. Tsoumakas, W. Groves, and I. Vlahavas, "Multi-target regression via input space expansion: treating targets as inputs," *Machine Learning*, vol. 104, no. 1, pp. 55–98, 2016.