2017 IEEE International Conference on Data Mining Workshops

Feature Selection in Learning Using Privileged Information

1st Rauf Izmailov
Vencore Labs
Basking Ridge, NJ, USA
rizmailov@vencorelabs.com

2nd Blerta Lindqvist
Department of Computer Science
Rutgers University
Piscataway, NJ, USA
b.lindqvist@rutgers.edu

3rd Peter Lin
Vencore Labs
Basking Ridge, NJ, USA
plin@vencorelabs.com

***Abstract*—The paper considers the problem of feature selection in learning using privileged information (LUPI), where some of the features (referred to as privileged ones) are only available for training, while being absent for test data. In the latest implementation of LUPI, these privileged features are approximated using regressions constructed on standard data features, but this approach could lead to polluting the data with poorly constructed and/or noisy features. This paper proposes a privileged feature selection method that addresses some of these issues. Since not many LUPI datasets are currently available in open access, while calibration of parameters of the proposed method requires testing it on a wide variety of datasets, a modified version of the method for traditional machine learning paradigm (i.e., without privileged features) was also studied. This lead to a novel mechanism of error rate reduction by constructing and selecting additional regression-based features capturing mutual relationships among standard features. The results on calibration datasets demonstrate the efficacy of the proposed feature selection method both for standard classification problems (tested on multiple calibration datasets) and for LUPI (for several datasets described in the literature).**

***Index Terms*—Learning systems, Support vector machines, Supervised learning, Knowledge representation, Knowledge engineering, Machine learning, Statistical analysis, Regression analysis.**

I. Introduction

The classical machine learning paradigm considers the following simple scheme: given a set of training examples, find, in a given set of functions, the one that approximates the unknown decision rule in the best possible way. In such a paradigm, Teacher does not play an important role (it only supplies classification labels). However, in human learning, the role of Teacher is much more sophisticated: along with labels of examples, Teacher provides students with explanations, comments, comparisons, metaphors, and so on.

This paper considers the model of learning that includes the so-called Intelligent Teacher, who supplies Student with intelligent (privileged) information during training session. This privileged information exists for almost any learning problem and this information can significantly accelerate the learning process. In the learning paradigm called *Learning Using Privileged Information (LUPI)*, Intelligent Teacher provides additional (privileged) information x^* about training example x at the training stage (when Teacher interacts with Student). The important point in this paradigm is that privileged information is *not* available at the test stage (when Student operates without supervision of Teacher).

This material is based upon work partially supported by AFRL and DARPA under contract FA8750-17-C-0121 and the work partially supported by AFRL under contract FA9550-15-1-0502. The views, opinions and/or findings expressed are those of the author and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

Formally, the classical paradigm of machine learning is described as follows [1], [2]: given a set of iid pairs (training data)

$$\begin{array}{c}(y_1, x_1), \ldots, (y_\ell, x_\ell),\\ x_i \in X, \quad y_i \in \{-1, +1\},\end{array} \tag{1}$$

generated according to a fixed but unknown probability measure $P(x, y) = P(y \mid x)P(x)$, find, in a given set of indicator functions $f(x, \alpha), \alpha \in \Lambda$, the function $y = f(x, \alpha_*)$ that minimizes the probability of incorrect classifications (incorrect values of $y \in \{-1, +1\}$). In this model, each vector $x_i \in X$ is a description of an example generated according to an unknown generator $P(x)$ of random vectors x_i, and $y_i \in \{-1, +1\}$ is its classification defined by Teacher according to an unknown conditional probability $P(y \mid x)$. The goal is to find the function $y = f(x, \alpha_*)$ that guarantees the smallest probability of incorrect classifications. That is, the goal is to find the function which minimizes the risk functional

$$R(\alpha) = \frac{1}{2} \int |y - f(x, \alpha)| dP(x, y) \tag{2}$$

in the given set of indicator functions $f(x, \alpha),\ \alpha \in \Lambda$ when the probability measure $P(x, y) = P(y \mid x)P(x)$ is unknown but training data (1) are given.

The LUPI paradigm [3], [4] describes a more complex model: given a set of iid triplets

$$\begin{array}{c}(y_1, x_1, x_1^*), \ldots, (y_\ell, x_\ell, x_\ell^*),\\ x_i \in X, \quad x_i^* \in X^*, \quad y_i \in \{-1, +1\},\end{array} \tag{3}$$

generated according to a fixed but unknown probability measure $P(x, x^*, y) = P(x^*, y \mid x)P(x)$, find, in a given set of indicator functions $f(x, \alpha), \alpha \in \Lambda$, the function $y = f(x, \alpha_*)$ that guarantees the smallest probability of incorrect classifications (2). In this model, each vector $x_i \in X$ is a description of an example generated according to an unknown generator $P(x)$ of random vectors x_i, and Intelligent Teacher generates both its label $y_i \in \{-1,+1\}$ and the privileged information x_i^* using some unknown conditional probability function $P(x_i^*, y_i \mid x_i)$.

In the LUPI paradigm, there is exactly the same goal of minimizing (2) as in the classical paradigm, i.e., to find the best classification function in the admissible set. However, during the training stage, there is more information, i.e., triplets (x, x^*, y) instead of pairs (x, y), than in the classical paradigm. The additional privileged information $x^* \in X^*$ belongs to space X^* which is, generally speaking, different from X.

The concept of LUPI was initially introduced in [3] and [4]; subsequent work targeted applications of LUPI methodology to a wide range of problems (see, for instance, [5], [6], [7], [8], [9]). Typically, privileged space represents information that is unavailable during test phase because it is either too expensive or too difficult to collect in timely manner.

The first implementation of the initial LUPI framework was presented in [10]. However, the SVM+ method proposed there was difficult to scale beyond a few hundred training samples. Recent papers [11], [12], [13] resolved the scalability issue by using the concept of *knowledge transfer* (generally described in [14]). In this paper, we consider this knowledge transfer in the LUPI framework, which is as scalable as any standard machine learning approach such as SVM or neural networks [15].

Although the problem of LUPI scalability is now solved, the nature of knowledge transfer method in LUPI raises another problem – namely, the problem of privileged feature selection. While the general feature selection problem is well studied in traditional machine learning (see, for instance, [16], [17], [18]), the LUPI framework imposes additional constraints that have to be taken into account. This problem is the main subject of this paper.

The paper is organized in the following way. In Section II, we describe the LUPI framework based on knowledge transfer. In Section III, we describe the problem of privileged feature selection that is present in this framework. In Section IV, we describe our method of executing privileged feature and present performance results both for LUPI framework and for traditional machine learning (for which, instead of privileged features, we consider derived features of the same type that are used in LUPI). We summarize our results in Section V, where we also outline potential next steps in this research.

II. LUPI with Knowledge Transfer

It is assumed that we are given a set of iid triplets

$$\begin{aligned}&(y_1, x_1, x_1^*), \ldots, (y_\ell, x_\ell, x_\ell^*),\\ &x_i \in X, \quad x_i^* \in X^*, \quad y_i \in \{-1,+1\},\end{aligned} \tag{4}$$

generated according to a fixed but unknown probability measure $P(x, x^*, y)$. Our training dataset consists of ℓ decision vectors $x_1, \ldots, x_\ell$ from n-dimensional decision space $X = R^n$ and corresponding ℓ privileged vectors $x_1^*, \ldots, x_\ell^*$ from m-dimensional privileged space $X^* = R^m$.

Specifically, for each feature $j = 1, 2, \ldots, m$, the following is done: using n-dimensional vectors $x_1, x_2, \ldots, x_\ell$ as explanatory variables and corresponding scalar values $(x_1^*)^j$, $(x_2^*)^j$, …, $(x_\ell^*)^j$ as response variables (upper indices denote vector coordinates), construct a (linear or nonlinear) regression function φ_j so that

$$\begin{cases} \varphi_j(x_1) = z_1^j \approx (x_1^*)^j \\ \varphi_j(x_2) = z_2^j \approx (x_2^*)^j \\ \qquad \cdots \\ \varphi_j(x_\ell) = z_\ell^j \approx (x_\ell^*)^j \end{cases}$$

for pairs $(x_1, x_1^*), \ldots, (x_\ell, x_\ell^*)$ from (4).

Various types of regression could be used for that purpose; in this paper, we use two of them: (1) linear regression; and (2) nonlinear kernel ridge regression which is constructed as an approximation of the regressed values with linear combination of radial basis functions (where parameters are selected using 6-fold cross-validation).

In the next step, we create, following the previously described framework of knowledge transfer, the modified training dataset, consisting of the concatenation of m-dimensional regression-based replacements of privileged vectors with the original decision data. As a result, our modified training data will form the matrix

$$\begin{pmatrix} y_1 & x_1^1 & \cdots & x_1^n & \varphi_1(x_1) & \cdots & \varphi_m(x_1) \\ y_2 & x_2^1 & \cdots & x_2^n & \varphi_1(x_2) & \cdots & \varphi_m(x_2) \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ y_\ell & x_\ell^1 & \cdots & x_\ell^n & \varphi_1(x_\ell) & \cdots & \varphi_m(x_\ell) \end{pmatrix}.$$

After that, we train an SVM algorithm on the modified set of $(n+m)$-dimensional vectors and construct the corresponding classification decision function F, which, when applied to any $(n+m)$-dimensional vector Z, produces the classification output $Y = F$(Z), where $Y \in \{-1,+1\}$.

The designed classification decision algorithm F can now be applied to any standard vector x from n-dimensional space R^n in following manner. First, we construct m scalar values

$$z^1 = \varphi_1(x), z^2 = \varphi_2(x), \ldots, z^m = \varphi_m(x).$$

using already constructed (during training) m regressions $\varphi_1, \varphi_2, \ldots, \varphi_m$. Then, we construct $(n+m)$-dimensional vector Z by concatenating these m scalar values with n-dimensional vector X:

$$Z = (x^1 \quad x^2 \quad \ldots \quad x^n \quad z^1 \quad z^2 \quad \ldots \quad z^m)$$

Finally, we apply the classification decision algorithm F to the constructed $(n+m)$-dimensional vector Z and obtain the classification label $Y = F$(Z), where $Y \in \{-1,+1\}$; this label Y is the desired classification of standard n-dimensional vector x.

The described procedure is illustrated in Fig. 1. The upper part of Fig. 1 shows the traditional machine learning approach, where both training and test datasets belong to the same space (and thus contain the same number of features), so the classification algorithm, trained on these features, is directly applicable to the test data. The lower part of Fig. 1 shows the knowledge transfer in the LUPI approach, where the training dataset contains, along with standard (blue) features, the privileged (red) ones. The knowledge transfer part of the LUPI algorithm then tries to *learn* these feature as functions (regressions) of standard ones, thus producing approximations (pink) of privileged (red) features. These approximations are then used for the test set, in order to augment standard features in the test set with approximations of the privileged ones. The actual learning algorithm is then trained and applied on datasets augmented with the constructed approximated features.

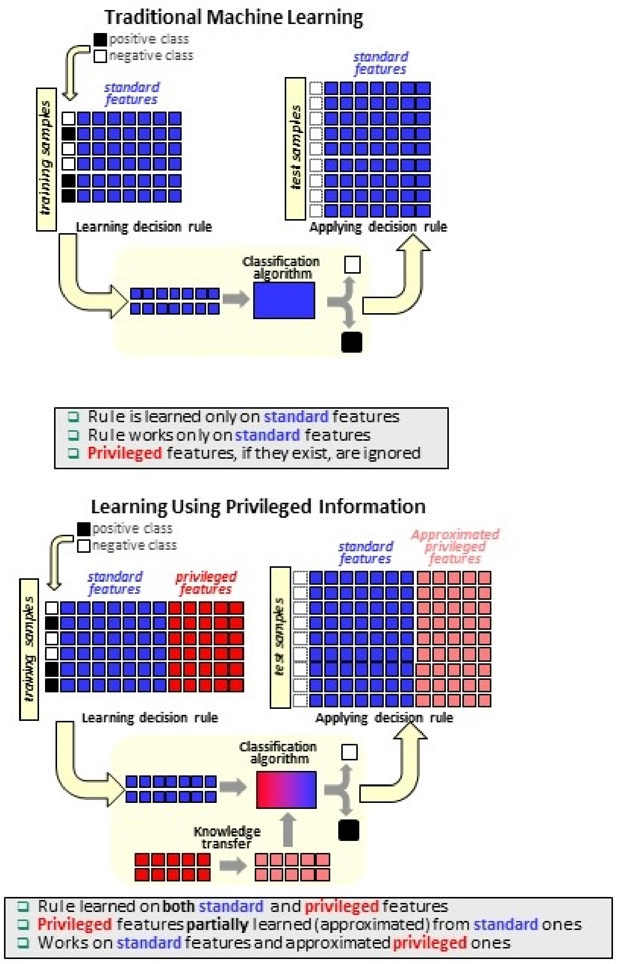
Traditional Machine Learning
positive class
negative class
standard features
training samples
test samples
standard features
Learning decision rule
Applying decision rule
Classification algorithm
Rule is learned only on standard features
Rule works only on standard features
Privileged features, if they exist, are ignored
Learning Using Privileged Information
positive class
negative class
standard features
privileged features
training samples
test samples
standard features
Approximated privileged features
Learning decision rule
Applying decision rule
Classification algorithm
Knowledge transfer
Rule learned on both standard and privileged features
Privileged features partially learned (approximated) from standard ones
Works on standard features and approximated privileged ones

Fig. 1. Traditional machine learning and knowledge transfer in LUPI.

The described procedure is greedy in the sense that it relies on approximating *all* privileged features through standard ones. In the next section, we explain why that could be a problem and outline our methodology for addressing it.

III. Privileged Feature Selection Problem

The traditional feature selection [16], [17], [18] focuses on a singe metric: relevance (expressed as correlation, mutual information, etc.) of a feature (or a group of features) for the classification label. In knowledge transfer LUPI, there is, in addition to that, the metric of quality of approximation of a privileged feature from standard ones.

Obviously, *both* of these metrics are important: if a privileged feature is very relevant for the label while being poorly approximated by standard features, its inclusion into LUPI framework would be equivalent to adding a noisy feature to the data. Similarly, if a privileged feature is well approximated by standard ones, while being not very relevant for the label (the adjective "privileged" does not, by itself, mean that the feature is useful), its inclusion would be also equivalent to adding a noisy feature. Therefore, a good privileged feature selection method should be able to capture *both* quantities.

One of the problems in designing and calibrating such a feature selection method is the relative scarcity of LUPI calibration problems, which can be explained by the fact that the scalable LUPI framework was introduced only recently and there is currently just a handful of LUPI problems available in the literature.

In contrast to that, there are significantly more (orders of magnitude more) calibration classification problems available through such machine learning repositories as [19] and [20]. This suggests the following approach towards designing and calibrating the privileged feature selection method.

We consider a standard classification problem with m standard features (without any privileged ones) and then apply the same knowledge transfer method employed by LUPI to its features. Specifically, for each feature f_i, where $i = 1, \dots, m$ we construct a regression φ_i of that feature based on complementary features $f_1, \dots, f_{i-1}, f_{i+1}, \dots, f_m$. We then treat these approximations in the same way as it is done in knowledge transfer LUPI: we augment standard features with these approximations for both training and test sets and then train the classification algorithm (in this paper, we use SVM) for this augmented dataset.

The described procedure is illustrated in Fig. 2. The upper part of the figure shows six features of a dataset and constructed regressions for each of these features. The lower part of the figure shows the resulting augmentation of the standard dataset consisting of actually *observed* or measured features with their *imputed* versions. Essentially, this provides dual views of the same feature, thus allowing the use of both data-driven observable and model-driven imputed versions of the same attribute as inputs to machine learning algorithms. The latter imputed version can also be viewed as a result of filtering of measurement errors in the observed version.

Although seemingly a futile exercise (newly constructed features would have exactly the same information content as the underlying ones), this method essentially (1) embeds the data into a higher-dimensional space by adding more derived

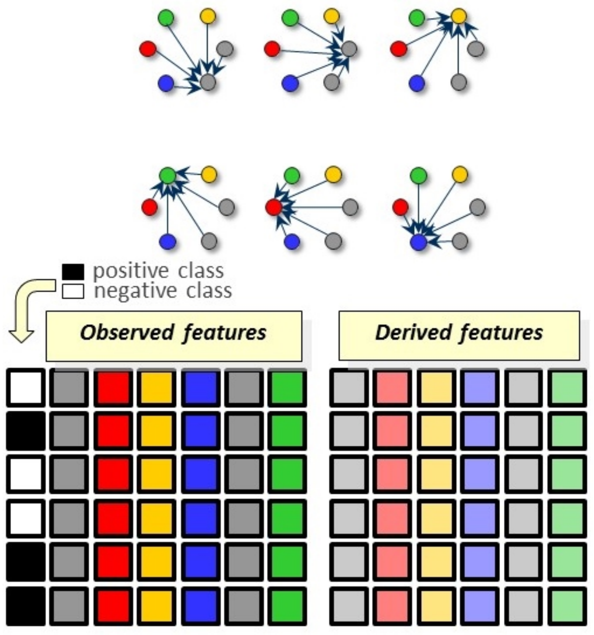

Fig. 2. Knowledge transfer within standard features.

features thus enriching the set of potential decision rules, while simultaneously (2) avoiding the curse of dimensionality by restricting the data into a lower-dimensional manifold (defined by said constructed relations) in that space.

It is now reasonable to assume that if we want to design and calibrate a privileged feature selection method, we can do it in a statistically reliable manner on a wide set of frequently used standard classification problems. As a result, besides being able to execute privileged feature selection for LUPI problems (where we transfer additional information from privileged space to the standard one), we would thus have a method of obtaining additional information (in the form of these mutual regressions) in the standard spaces, which, as we show in the next section, can also improve classification performance for standard non-LUPI problems.

IV. Privileged Feature Selection Method

As mentioned in the previous section, we consider two types of regressions: linear and nonlinear. In this paper, nonlinear regression denotes kernel ridge regression with RBF kernel. Thus, for each (observed) feature in the original N-dimensional dataset, we create two new (derived) features, each constructed as either a linear or a nonlinear regression of the complementary $N-1$ features in the original dataset.

The key question is which of these derived features, if any, should be augmented to the original dataset. As explained in the previous section, we are looking for each feature to be both well approximated by the corresponding regressions (which should be estimated using some *approximation metric*) and for each feature to be relevant to the classification task (which should be estimated using some *relevance metric*).

In this paper, we use the same type of computation for both approximation metric and relevance metric. Specifically, we use the adjusted mutual information score $AMI(U,V)$ [21] for measuring similarity between two sets of values (features) U and V. It is calculated as

$$AMI(U,V) = \frac{MI(U,V) - E(MI(U,V))}{\max(H(U),H(V)) - E(MI(U,V))},$$

where H, MI, and E denote, respectively, entropy, mutual information, and expectation. We used the implementation provided in the scikit-learn [22] module. The AMI score is an adjustment of the mutual information score to account for chance. It is symmetric and returns a value of 1 when the two sets are identical. Random values have an expected AMI metric value of 0, thus the computed values can be negative.

Using the AMI score, we compute the approximation metric C_1 as the AMI score between the regressed feature V (derived one) and the original feature U (observed one). We also compute the relevance metric C_2 as the AMI score between the regressed feature V (derived one) and the labels, viewed as a special feature B. Since we want both metrics to be satisfactory, we only include the regressed (derived) feature into the set of augmented features if the following two conditions are satisfied:

$$\begin{cases} \text{Approximation condition}: & C_1 = AMI(U,V) > 0.5 \\ \text{Relevance condition}: & C_2 = AMI(B,V) < 0.5 \end{cases}$$

In order to test and calibrate the proposed algorithm, we conducted the following series of experiments illustrated in Table I. First, we have selected a number of diverse classification problems from UCI Machine Learning Repository [19] and OpenML [20] that are shown in the first column of Table I. For each of the selected problems, we created 20 pairs of training and test subsets by randomly partitioning the given dataset. For some datasets, we used around 75% of the data for training and the remaining 25% for test; in other (larger) datasets, we selected training size in the vicinity of several hundred, leaving all the other data for test. Specific training and test data sizes for each of the selected problems are listed in the second and third columns of Table I, the number of features is shown in the fourth column.

The next three columns of Table I show performance results (error rates) averaged over 20 pairs of training and test sets for the following three algorithms:

- SVM: SVM algorithm with RBF kernel;
- SVM-L: SVM algorithm with RBF kernel on the data augmented with those derived features that were constructed using linear regressions and satisfying both approximation and relevance conditions.
- SVM-LN: SVM algorithm with RBF kernel on the data augmented with those derived features that were constructed using either linear or nonlinear regressions and satisfying both approximation and relevance conditions.

Since both SVM-L and SVM-LN operate on augmented sets of features, the next two columns of Table I, labeled as Cols-L and Cols-LN, show how many derived features, on average, satisfy both approximation and relevance conditions for the corresponding algorithms.

The last column shows the best error improvement result (if it indeed improves), defined as the relative decrease of error rate. In most cases, the best error improvement is achieved with SVM-LN.

The results in Table I prompt the following observations:

- Performance improves (error rate decreases) for about 70% of the datasets where regressed columns are added - for a few of them quite significantly (almost 38% have error improvement higher than 3% and over 16% have error improvement higher than 10%). This suggests the value of constructing regressed features that can obviously capture some useful structural information (relations among features) and expose it to classification algorithm with positive results.
- Performance improves (error rate decreases) just a little for a number of considered cases – that is not surprising, since the structural information provided by constructed derived features, may already exist in the original ones, so no significant improvement is achieved.
- Importantly, performance almost never deteriorates (error rate does not increase) for the considered datasets (for one dataset, namely, "magic telescope", where it increases from by 0.01%, from 15.95% to 15.96%). Therefore, the combination of the proposed approximation and relevant conditions can indeed add value (if possible) without jeopardizing the default performance (achieved on the datasets with only the original features).
- The performance of SVM-LN is in most cases better or equal to that of SVM-L, which is yet another indicator that proposed approximation and relevance conditions are capable of decreasing error rate without jeopardizing existing performance.

We then applied the developed feature selection algorithm to LUPI. As mentioned in Section III, the main problem here is that there are no public repositories of calibration data with privileged features. Researchers in that area usually procure their own data (not shared in the public domain), which is a significant time-consuming effort (after all, privileged features are those that are usually too difficult or too expensive to obtain during in the course of usual procedures). As a result, we looked into ways to calibrate privileged feature selection algorithm by using open source datasets with features that can be reasonably partitioned into standard and privileged ones.

In order to achieve that, we looked into those partitions that engender a reasonable performance gap between training on standard features and training on all (standard and privileged features). Two examples are illustrated in Table II: the dataset Ionosphere [19] (where features 5, 6, 21, 22 were assigned as privileged) and the dataset kc2 [27] (where features 15, 16, 17, 18, 19, 20, 21 were assigned as privileged). In order to test and calibrate the proposed algorithm for these two datasets, we conducted the following series of experiments. For both problems, we created 20 pairs of training and test subsets by randomly partitioning the given dataset. For each dataset, we used around 75% of the data for training and the remaining 25% for test. Specific training and test data sizes for each of the selected problems are listed in the second and third columns of Table II; the numbers of all features and privileged ones are shown in the fourth and fifth columns, respectively.

The next five columns of Table II show performance results (error rates) averaged over 20 pairs of training and test sets for the following five algorithms that we compared here:

- SVMstd: SVM algorithm with RBF kernel on standard features (i.e., on those features that are not privileged);
- SVMprv: SVM algorithm with RBF kernel on both standard and privileged features (i.e., on all features);
- L-LUPI: LUPI algorithm using linear regressions of privileged features.
- NL-LUPI: LUPI algorithm using nonlinear regressions of privileged features.
- sel-LUPI: LUPI algorithm with feature selection proposed above.

The last three columns of Table II show LUPI results in the form of LUPI-specific performance gain, which we define following its definition in [15]. It is based on the observation that the error rates of LUPI are supposed to be between the corresponding SVMstd and SVMpriv. In other words, if the error rate of the SVMstd algorithm on standard features is B, while the error rate of the algorithm SVMpriv on all features is C, the error rate A of LUPI should satisfy the bounds $C < A < B$. So LUPI gain is evaluated by computing the metric $(B - A)/(B - C)$, which describes how much of the performance gap $B - C$ can be recovered by LUPI.

As Table II illustrates, the proposed privileged feature selection algorithm successfully achieves higher LUPI gains than either of the current LUPI implementations (with linear and nonlinear regressions).

V. Conclusions

We have considered the novel problem of privileged feature selection in LUPI and proposed a robust method for its solution. We have calibrated a version of that method for non-LUPI scenario, thus also obtaining a novel method of constructing and selecting derived features in standard classification problems, yielding some performance improvement (reduction of error rate) for many of them. We then applied this method to several LUPI problems described in the literature, yielding reduction of error rate as well.

As next steps, we plan to investigate both additional regression mechanisms for constructing approximations of privileged features and more diverse metrics and mechanisms for their selection.

References

[1] V. Vapnik, The nature of statistical learning theory. New York: Springer-Verlag, 1995.

[2] V. Vapnik, Statistical learning theory. New York: John Wiley & Sons, 1998.

[3] V. Vapnik, Estimation of dependencies based on empirical data (2nd Edition). New York, NY: Springer, 2006.

[4] V. Vapnik and A. Vashist, "A new learning paradigm: learning using privileged information," Neural Networks, vol. 22, pp. 546–557, 2009.

TABLE I
DERIVED FEATURE SELECTION

Datasets	***Train***	***Test***	***Features***	***SVM error***	***SVM-L error***	***SVM-LN error***	***Cols-L***	***Cols-LN***	***Improvement***
Collins [20]	375	125	22	25.56%	20.12%	18.60%	2.00	4.00	27.23%
Synthetic_control [23]	480	120	60	0.79%	0.71%	0.58%	31.70	64.00	26.32%
mu284 [20]	213	71	10	3.31%	3.31%	2.82%	0.45	0.90	14.89%
fl2000 [20]	50	17	15	20.29%	17.94%	19.71%	2.40	4.80	11.59%
Prnn-synth [20]	187	63	2	14.13%	14.05%	12.62%	2.00	4.00	10.67%
Wilt [24]	500	4339	5	2.65%	2.50%	2.45%	1.40	2.80	7.55%
Meta-ensembles [20]	62	12	62	41.67%	41.67%	39.17%	2.65	5.85	6.00%
fri_c0_100_5 [20]	80	20	5	13.50%	12.75%	13.50%	5.00	10.00	5.56%
Cloud [20]	81	27	7	46.48%	45.74%	44.44%	1.00	2.00	4.38%
Spectrometer [20]	398	133	101	45.04%	43.91%	43.27%	92.90	185.80	3.92%
Sonar [25]	187	21	60	13.10%	12.38%	12.62%	1.55	3.10	3.66%
ar4 [20]	90	17	29	16.47%	15.88%	16.47%	0.80	1.60	3.57%
Glass [26]	160	54	10	8.98%	8.80%	8.70%	0.35	0.85	3.12%
Analcatdata-wildcat [20]	130	33	5	22.88%	22.27%	22.73%	1.70	3.20	2.65%
pc1 [27]	831	278	21	6.83%	6.69%	6.69%	2.95	2.90	2.11%
Pasture [20]	30	6	22	40.00%	39.17%	39.17%	7.80	16.15	2.08%
kc2 [27]	417	105	21	16.95%	17.90%	16.62%	4.15	8.30	1.97%
kc1 [27]	500	1609	21	15.28%	15.24%	15.18%	3.85	6.10	0.67%
chscase_census [20]	300	100	6	41.85%	41.60%	41.60%	3.30	6.60	0.60%
Magic telescope [20]	700	18320	10	15.95%	15.90%	15.97%	2.35	4.50	0.30%
Pollen [20]	500	3348	5	50.58%	50.54%	50.45%	2.50	5.00	0.26%
Tamilnadu-electricity [20]	700	45081	3	32.46%	32.44%	32.39%	1.75	3.50	0.22%
Disclosure [28]	496	166	3	44.43%	44.34%	44.43%	3.00	6.00	0.20%
Steel plates fault [20]	500	14	33	0.14%	0.16%	0.12%	2.15	4.30	0.13%
Mushroom [19]	500	7624	21	34.58%	34.58%	34.58%	1.00	2.00	
Connect-4 [20]	500	67057	42	28.79%	28.79%	28.79%	0.70	1.40	
Ada-agnostic [20]	500	4062	48	21.94%	21.94%	21.94%	3.60	7.20	
Ecoli [20]	252	84	7	16.90%	16.90%	16.90%	0.25	0.50	
Kr-vs-kp [20]	500	2696	36	4.07%	4.07%	4.07%	1.20	2.40	
Triazines [20]	139	47	60	28.72%	28.72%	28.72%	2.30	4.30	
Datatrieve [27]	110	20	8	9.75%	9.75%	9.75%	0.05	0.05	
Meta-batchincremental [20]	62	12	62	35.00%	35.00%	35.00%	2.70	5.95	

TABLE II
PRIVILEGED FEATURE SELECTION

Datasets	***Train***	***Test***	***All Features***	***Privileged Features***	***SVMstd error***	***SVMpriv error***	***L-LUPI gain***	***NL-LUPI gain***	***sel-LUPI gain***
Ionosphere [19]	263	88	34	4	5.91%	4.89%	55.88%	45.10%	89.21%
kc2 [27]	392	130	21	7	16.84%	16.03%	59.26%	65.43%	81.48%

[5] 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, pp. 1086–1098, 2013.

[6] R. Ilin, S. Streltsov, and R. Izmailov, "Learning with privileged Information for improved target classification," International Journal of Monitoring and Surveillance Technologies Research, vol. 2(3), pp. 5–66, 2014.

[7] B. Ribeiro, C. Silva, N. Chen, A. Vieirac, and J. Carvalho das Nevesd, "Enhanced default risk models with SVM+," Expert Systems with Applications, vol. 39, pp. 10140–10152, 2012.

[8] V. Sharmanska and C. Lampert, "Learning to rank using privileged information," in 2013 IEEE International Conference on Computer Vision (ICCV), pp.825–832, 2013.

[9] H. Yang and I. Patras, "Privileged information-based conditional regression forest for facial feature detection," in 2013 IEEE International Conference on Automatic Face and Gesture Recognition, pp. 1–6, 2013.

[10] D. Pechyony, R. Izmailov, A. Vashist, and V. Vapnik, "SMO-style algorithms for learning using privileged information," in 2010 International Conference on Data Mining, pp. 235–241, 2010.

[11] V. Vapnik and R. Izmailov, "Learning using privileged information: similarity control and knowledge transfer," Journal of Machine Learning Research. vol. 16, pp. 2023–2049, 2015.

[12] V. Vapnik and R. Izmailov, "Learning with intelligent teacher: similarity control and knowledge transfer," in Statistical Learning and Data Sciences. LNAI 9047, pp. 3–32. London: Springer-Verlag, 2015.

[13] R. Ilin, R. Izmailov, Y. Goncharov, and S. Streltsov, "Fusion of privileged features for efficient classifier training," in Proceedings of 19th International Conference on Information Fusion, pp. 1–8, 2016.

[14] R. Brachman and H. Levesque, Knowledge representation and reasoning. San Francisco, CA: Morgan Kaufmann, 2004.

[15] V. Vapnik and R. Izmailov, "Knowledge transfer in SVM and neural networks," Annals of Mathematics and Artificial Intelligence, pp. 1-17, 2017.

[16] J. Liu and H. Motoda,Computational Methods of Feature Selection. Boca Raton, FL: Chapman & Hall/CRC, 2008.

[17] I. Guyon and A. Elisseeff,"An introduction to variable and feature selection," Journal of Machine Learning Research, vol. 3, pp. 1157–1182, 2003.

[18] K. Torkkola, "Feature extraction by non-parametric mutual information maximization," Journal of Machine Learning Research, vol. 3, pp. 1415–1438, 2003.

[19] M. Lichman, "UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]," Irvine, CA: University of California, School of Information and Computer Science, 2013.

[20] J. Vanschoren, J. van Rijn, B. Bischl, Bernd, and L. Torgo, "OpenML: Networked Science in Machine Learning," SIGKDD Explorations, vol. 15, pp. 49-60, 2013.

[21] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for

clusterings comparison: variants, properties, normalization and correction for chance," Journal of Machine Learning Research, vol. 11, pp. 2837–2854, 2010.

[22] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[23] D. Pham and A. Chan, "Control chart pattern recognition using a new type of self-organizing neural network," Proceedings of the Institution of Mechanical Engineers, vol. 212, pp. 115–127, 1998.

[24] B. Johnson, R. Tateishi, and N. Hoan, "A hybrid pansharpening approach and multiscale object-based image analysis for mapping diseased pine and oak trees," International Journal of Remote Sensing, vol.34, pp. 6969–6982, 2013.

[25] T. Sejnowski and R. Gorman, "Analysis of hidden units in a layered network trained to classify sonar targets," Neural Networks, vol. 1, pp. 75–89, 1988.

[26] I. Evett and E. Spiehler, "Rule induction in forensic science," Central Research Establishment, Home Office Forensic Science Service, 1987.

[27] J. S. Shirabad and T.J. Menzies, "The PROMISE Repository of software Engineering Databases," 2005. url = "http://promise.site.uottawa.ca/SERepository"

[28] S. Fienberg, U. Makov, and A. Sanil, "A Bayesian approach to data disclosure: optimal intruder behavior for continuous data," Journal of Official Statistics, vol. 13, pp. 75–89, 1997.