Learning with Intelligent Teacher

Vladimir Vapnik[1,2] and Rauf Izmailov[3](✉)

[1] Columbia University, New York, NY, USA
vladimir.vapnik@gmail.com
[2] AI Research Lab, Facebook, New York, NY, USA
[3] Applied Communication Sciences, Basking Ridge, NJ, USA
rizmailov@appcomsci.com

Abstract. The paper considers several topics on learning with privileged information: (1) general machine learning models, where privileged information is positioned as the main mechanism to improve their convergence properties, (2) existing and novel approaches to leverage that privileged information, (3) algorithmic realization of one of these (namely, knowledge transfer) approaches, and its performance characteristics, illustrated on simple synthetic examples.

Keywords: Intelligent teacher · Privileged information · Similarity control · Knowledge transfer · Knowledge representation · Frames · Support vector machines · SVM+ · Classification · Learning theory · Kernel functions · Regression

1 Introduction

The classical machine learning paradigm considers a 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.

In human learning, however, the role of Teacher is important: along with 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

V. Vapnik—This material is based upon work partially supported by AFRL and DARPA under contract FA8750-14-C-0008 and the work partially supported by AFRL under contract FA9550-15-1-0502. Any opinions, findings and/or conclusions in this material are those of the authors and do not necessarily reflect the views of AFRL and DARPA.

© Springer International Publishing Switzerland 2016
A. Gammerman et al. (Eds.): COPA 2016, LNAI 9653, pp. 3–19, 2016.
DOI: 10.1007/978-3-319-33395-3_1

important point in this paradigm is that privileged information is *not* available at the test stage (when Student operates without supervision of Teacher). LUPI was initially introduced in [15,16]; subsequent work targeted various implementation issues of this paradigm [9] and its applications to a wide range of problems [3,4,10,12,19].

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

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

generated according to a fixed but unknown probability measure $P(x, y) = P(y|x)P(x)()$, 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 $P(y|x)$. 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 $P(x, y) = P(y|x)P(x)$ is unknown but training data (1) are given.

The LUPI paradigm describes a more complex model: given a set of iid triplets

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

generated according to a fixed but unknown probability measure $P(x, x^*, y) = P(x*, y|x)P(x)$, 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 $P(x_i^*, y_i|x_i)$.

In the LUPI paradigm, we have 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, we have more information, i.e., we have triplets (x, x^*, y) instead of pairs (x, y) as in the classical paradigm. The additional information $x^* \in X^*$ belongs to space X^* which is, generally speaking, different from X.

The paper is organized in the following way. In Sect. 2, we outline general models of information theory and their relation to models of learning. In Sect. 3, we explain how privileged information can significantly accelerate the rate of

learning (i.e., accelerate the convergence) when the notion of classical learning model is expanded appropriately to incorporate privileged information. In Sect. 4, we argue that structures in the space of privileged information reflect more fundamental properties of learning and thus can potentially improve the performance of learning methods even further. We outline a general *knowledge transfer* framework for realization of that improvement in Sect. 5. In Sect. 6, we present some specific algorithms implementing elements of that framework and illustrate their various properties on synthetic examples We conclude with Sect. 7, in which we summarize our results and outline potential next steps in this research.

2 Brute Force and Intelligent Models

In this section, we show how the general setting of machine learning problems creates a background for introduction of the concept of privileged information.

According to Kolmogorov [7], there exist three categories of integer numbers.

1. **Ordinary numbers:** those numbers n that we use in our everyday life. For simplicity, let these numbers be between 1 and one billion.
2. **Large numbers:** those numbers N that are between one billion and 2^n (where n belongs to the category of ordinary numbers).
3. **Huge numbers:** those numbers H that are greater than $2^N = 2^{2^n}$ (where N belongs to the category of large numbers).

Kolmogorov argued that the ordinary integers n correspond to the number of items we can handle realistically, say the number of examples in a learning problem. We cannot realistically handle large numbers (say large number of examples in a learning problem), but we can still treat them efficiently in our theoretical reasoning using mathematics; however, huge numbers are beyond our reach. In this paper, we describe methods that potentially might operate in huge sets of functions. In contrast to methods based on mathematical models and suitable for large numbers (which we call "brute force" methods), these methods include intelligent agents and thus can be viewed as "intelligent methods".

Basic Shannon Model. Suppose that our goal is to find one function among large number N of different functions by making ordinary number of queries that return the reply "yes" or "no" (thus providing one bit of information). Theoretically, we can find the desired function among N functions by making n queries, where $n = \log_2 N$ (for simplicity, we assume that N is an integer power of 2). Indeed, we can split the set of N functions into two subsets and make query to which subset the desired function belongs: to the first one (reply $+1$) or to the second one (reply -1). After obtaining the reply from the query, we can remove the subset which does not contain the desired function, split the remaining part into two subsets, and continue in the same fashion, removing half of the remaining functions after each reply. So after $n = \log_2 N$ queries we will

find the function. It is easy to see that one cannot guarantee that it is possible to find the desired function by making less than

$$n = \log_2 N = \frac{\ln N}{\ln 2} \tag{4}$$

queries. This also means that one cannot find one function from the set of huge number $H = 2^N$ of functions: this would require to make too many (namely N) queries, which is unrealistic.

Basic Model using Language of Learning Theory. Let us repeat this reasoning for pattern recognition model. Suppose that our set $y = f(x, \alpha_t)$, $t = 1, \ldots, N$ is a finite set of binary functions in $x \in R^n$. That is, $f(x, \alpha_t) \in \{-1, +1\}$. Suppose that we can construct such vector $x_1 \in R^n$ that half of functions take value $f(x_1, \alpha^*_{t_1}) = +1$ and another half take value $f(x_1, \alpha_{t_1}) = -1$. Then the query for the label of vector x_1 provides the first element of training data (x_1, y_1). As before, we remove half of the functions that replied $-y_1$ and continue this process. After collecting at most $n = \ln N / \ln 2$ elements of training examples, we obtain the desired function.

First Modification of the Learning Model. To find the function in framework of basic model requires solution of a difficult problem: on any step of the procedure to find a vectors x_i that splits the remaining set of functions into two equal parts (suppose that such a vector exists). To simplify our model, consider the situation where vectors x are results of random iid trial with a fixed (but unknown) probability measure $p(x)$, and for any x we can query for its label y. After each query, we remove the functions that return $-y$ on x. The main problem for this model is to determine how many queries about labels one has to make[1] to find the function that is ε-close to the desired one with probability $1-\eta$ (recall that the desired function is any function among those that do not make errors, and ε-closeness is defined with respect to measure $p(x)$). The answer to this problem constitutes a special case of the VC theory [13,14]: the number of the required queries is at most

$$\ell = \frac{\ln N - \ln \eta}{\varepsilon}. \tag{5}$$

This expression differs from bound (4) by a constant: $(\varepsilon)^{-1}$ instead of $(\ln 2)^{-1}$. After this number of queries, any function in the remaining set is ε-close to the desired one. This bound cannot be improved.

Second Modification of the Basic Model. So far, we considered the situation when the set of N functions includes the one that does not makes errors.

[1] In other words, how large should be the number ℓ of training examples $(x_1, y_1), ..., (x_\ell, y_\ell)$.

Now we relax this assumption: any function in our set of N functions can make errors. Our problem is to find the function than provides the smallest probability of error with respect to probability measure $p(x)$.

Now we cannot use the method for choosing the desired function defined in the first model: removing from the consideration the functions from the set that disagree with classification of query. We will use another (a more general) algorithm which selects such function among N of them that make the smallest number of disagreements with the query reply (i.e., minimizes the empirical loss) on the training set

$$(x_1, y_1), \ldots, (x_\ell, y_\ell).$$

In order to guarantee that we will select an ε-close function to the best in the set of N elements with probability $1 - \eta$, one has to make at most

$$\ell = \frac{\ln N - \ln \eta}{\varepsilon^2}$$

queries. Again, in this modification, the main term $\ln N$ remains the same but constant $(\varepsilon)^{-2}$ is different from the constant in (5). This bound cannot be improved.

Third Modification (VC Model). Consider now the set of functions $f(x, \alpha)$, $\alpha \in \Lambda$ with infinite number of elements. Generally speaking, in this situation one cannot guarantee that it is possible to obtain a good approximation even if we have a large number of training examples. Recall that in the more simple situation with a set that contains finite but huge number of functions $H = 2^{2^n}$, one needs 2^n examples, which is far beyond our reach. Nevertheless, if infinite set of functions has finite VC dimension $VCdim$, then ε-close solution can be found with probability $1 - \eta$ using at most

$$\ell = \frac{VCdim - \ln \eta}{\varepsilon}$$

observations, if the desired function does not make errors; otherwise, if errors are allowed,

$$\ell = \frac{VCdim - \ln \eta}{\varepsilon^2}$$

observations are required. Note that this bound matches the form of bound (5), where the value of VC dimension replaces the logarithm of the number of functions in the set. This bound cannot be improved.

The finiteness of the VC dimension of the set of functions defines the necessary and sufficient conditions of learnability (consistency) of empirical risk minimization method. This means that VC dimension characterizes not just the *quantity* of elements of the set, it characterizes something else, namely, the *measure of diversity* of the set of functions: the set of functions must be not too diverse.

The structural risk minimization principle that uses structure on the nested subsets of functions with finite VC dimension (defined on the sets of functions

which closure can have infinite VC dimension) guarantees convergence of risk to the best possible risk for this structure [13,14].

To summarize, we have outlined the best bounds for general machine learning models and stated that they cannot be improved. In other words, in order to improve these bounds, the models themselves will have to be changed. The specific model change that we are concerned with in this paper is provided by the notion of privileged information, which is described and explored in the subsequent sections.

3 Privileged Information As Learning Acceleration

The learning models described in the previous section can be solved by different methods. In particular, SVM algorithms with universal kernels realize structural risk minimization method and thus are universally consistent. This means that the VC theory completely solves the problem of learning from examples providing not only the necessary and sufficient conditions of learnability but also an effective practical algorithm for machine learning. The rate described by this theory cannot be improved essentially (without additional information).

The intriguing question in VC theory was why the number of examples one needs to construct ε-close hyperplane in separable case (when training data can be separated without errors) and unseparable case (when training data cannot be separated without errors) vary so much in their corresponding constants (ε^{-1} and ε^{-2}).

For SVM algorithm, this effect can be explained by noticing that, in the separable case, using ℓ examples one has to estimate n parameters w of hyperplane, while in the non-separable case, one has to estimate, along with parameters n of hyperplane w, the additional ℓ values of slacks (making the total number of parameters to be estimated larger than number of examples). This, however, can addressed by a special SVM+ algorithm within the LUPI framework [15,16]. In that framework, Intelligent Teacher supplies Student with triplets

$$(x_1, x^*, y_1), \ldots, (x_\ell, x^*_\ell, y_\ell)$$

where $x_i \in X^*$, whereas, in the classical setting of the problem, Student uses training pairs

$$(x_1, y_1), \ldots, (x_\ell, y_\ell)$$

where vector $x_i \in X$ is generated by the generator of random events $p(x)$ and Teacher supplies Student with the label $y_i \in \{-1, +1\}$. In contrast to classical setting, in the LUPI paradigm, Intelligent Teacher supplies Student with triplets (x_i, x^*_i, y_i) where vector $x^*_i \in X^*$ and label y are generated by conditional probability $p(x^*, y|x)$. Formally, by providing both vector $x^* \in X^*$ and label y_i for any example x_i, Intelligent Teacher can supply Student with *more than one bit of information*, so the rate of convergence can be faster.

Indeed, as was shown in [15,16], this SVM+ approach in LUPI can improve the constant from ε^{-2} to ε^{-1}. The recent LUPI papers [17,18] introduced more

important approaches that could be potentially used for further improvement of convergence. In order to use such mechanisms effectively, Intelligent Teacher has to possess some knowledge that can describe physical model of events better than x. In the subsequent sections, we describe these ideas in greater detail.

4 Space of Privileged Information

Let us suppose that Intelligent Teacher has some knowledge about the solution of a specific pattern recognition problem and would like to transfer this knowledge to Student. For example, Teacher can reliably recognize cancer in biopsy images (in a pixel space X) and would like to transfer this skill to Student.

Formally, this means that Teacher has some function $y = f_0(x)$ that distinguishes cancer ($f_0(x) = +1$ for cancer and $f_0(x) = -1$ for non-cancer) in the pixel space X. Unfortunately, Teacher does not know this function explicitly (it only exists as a neural net in Teacher's brain), so how can Teacher transfer this construction to Student? Below, we describe a possible mechanism for solving this problem; we call this mechanism *knowledge transfer*.

Suppose that Teacher believes in some theoretical model on which the knowledge of Teacher is based. For cancer model, he or she believes that it is a result of uncontrolled multiplication of the cancer cells (cells of type B) which replace normal cells (cells of type A). Looking at a biopsy image, Teacher tries to generate privileged information that reflects his or her belief in development of such s process; Teacher can describe the image as:

Aggressive proliferation of cells of type B into cells of type A.

If there are no signs of cancer activity, Teacher may use the description

Absence of any dynamics in the of standard picture.

In uncertain cases, Teacher may write

There exist small clusters of abnormal cells of unclear origin.

In other words, Teacher is developing a specialized language that is appropriate for description x_i^* of cancer development using the model he believes in. Using this language, Teacher supplies Student with privileged information x_i^* for the image x_i by generating training triplets

$$(x_1, x_1^*, y_1), \ldots, (x_\ell, x_\ell^*, y_\ell). \tag{6}$$

The first two elements of these triplets are descriptions of an image in two languages: in language X (vectors x_i in pixel space), and in language X^* (vectors x_i^* in the space of privileged information), developed for Teacher's understanding of cancer model.

Note that the language of pixel space is universal (it can be used for description of many different visual objects; for example, in the pixel space, one can

distinguish between male and female faces), while the language used for describing privileged information is very specialized: it reflects just a model of cancer development. This has an important consequence: the set of admissible functions in space X has to be rich (has a large VC dimension), while the set of admissible functions in space X^* may be not rich (has a small VC dimension).

One can consider two related pattern recognition problems using triplets (6):

1. The problem of constructing a rule $y = f(x)$ for classification of biopsy in the pixel space X using data

$$(x_1, y_1), \ldots, (x_\ell, y_\ell). \tag{7}$$

2. The problem of constructing a rule $y = f^*(x^*)$ for classification of biopsy in the space X^* using data

$$(x_1^*, y_1), \ldots, (x_\ell^*, y_\ell). \tag{8}$$

Suppose that language X^* is so good that it allows to create a rule $y = f_\ell^*(x^*)$ that classifies vectors x^* corresponding to vectors x with the same level of accuracy as the best rule $y = f_\ell(x)$ for classifying data in the pixel space.[2]

Since the VC dimension of the admissible rules in a special space X^* is much smaller than the VC dimension of the admissible rules in the universal space X and since, the number of examples ℓ is the same in both cases, the bounds on error rate for the rule $y = f_\ell^*(x^*)$ in X^* will be better[3] than those for the rule $y = f_\ell(x)$ in X. That is, generally speaking, the classification rule $y = f_\ell^*(x^*)$ will be more accurate than classification rule $y = f_\ell(x)$.

As a result, the following problem arises: how one can use the knowledge of the rule $y = f_\ell^*(x^*)$ in space X^* to improve the accuracy of the desired rule $y = f_\ell(x)$ in space X? A general framework for that is outlined in the next section.

5 Knowledge Transfer from Privileged Space

As already described, knowledge transfer approach deals with iid training examples generated by some unknown generator $P(x), x \in X$ and Intelligent Teacher who supplies vectors x with information $(x^*, y|x)$ according to some (unknown) *Intelligent generator* $P(x^*, y|x), x^* \in X^*, y \in \{-1, +1\}$, forming training triplets

$$(x_1, x_1^*, y_1), \ldots, (x_\ell, x_\ell^*, y_\ell). \tag{9}$$

[2] The rule constructed in space X^* cannot be better than the best possible rule in space X, since all information originates in space X.

[3] According to VC theory, the guaranteed bound on accuracy of the chosen rule depends only on two factors: frequency of errors on training set and VC dimension of admissible set of functions.

Consider two pattern recognition problems in decision and privileged spaces:

1. *Pattern recognition problem defined in space*X: Using data, $(x_1, y_1), \ldots, (x_\ell, y_\ell)$, find in set of functions $f(x, \alpha), \alpha \in \Lambda$ the rule $y = \text{sgn}\{f_\ell(x)\}$ that minimizes the probability of test errors (in space X).
2. *Pattern recognition problem defined in space* X^*: Using data, $(x_1^*, y_1), \ldots, (x_\ell^*, y_\ell)$, find in set of functions $f^*(x^*, \alpha^*), \alpha^* \in \Lambda^*$ the rule $y = \text{sgn}\{f_\ell^*(x^*)\}$ that minimizes the probability of test errors (in space X^*).

Suppose that, in space X^*, one can find a rule $y = \text{sgn}\{f_\ell^*(x^*)\}$ that is, with probability $1 - \eta$, is better than the corresponding rule $y = \text{sgn}\{f_\ell(x)\}$ in space X. Also, suppose that we are looking for our rule in the form

$$f_\ell^*(x^*) = \sum_{i=1}^{\ell} y_i \alpha_i K_i^*(x^*) + b^*, \tag{10}$$

where $\alpha_i^*,\ i = 1, \ldots, \ell$ and b^* are parameters, and K_i are some functions in X^*. The question is whether the knowledge of a good rule (10) in space X^* can be used to find a good rule

$$s = f_\ell(x) = \sum_{i=1}^{\ell} y_i \alpha_i K(x_i, x) + b \tag{11}$$

in space X.

As was described in the previous section for the problem of cancer diagnostics, since pixel space X is universal and space of descriptions X^* reflects just the model of cancer development[4], the VC dimension of admissible set of functions in X space has to be much bigger than VC dimension of admissible set of functions in X^*. Therefore, with probability $1 - \eta$, the guaranteed quality of the rule constructed from ℓ examples in space X^* will be better than the quality of the rule constructed from ℓ examples in space X. That is why a transfer of a rule from space X^* into space X can be helpful.

In order to describe knowledge transfer, consider two fundamental concepts of knowledge representation used in Artificial Intelligence [1]:

1. Frames (fragments) of the knowledge.
2. Structural connections of the frames (fragments) in the knowledge.

The actual realization of frames and structures of knowledge can be done using different methods. For example, we can call the *frames in the knowledge* the smallest number of the vectors $u_1^* \ldots, u_m^*$ from space X^* that can approximate[5] the main part of the rule (10):

$$f_\ell^*(x^*) - b^* = \sum_{i=1}^{\ell} y_i \alpha_i^* K^*(x_i^*, x^*) \approx \sum_{k=1}^{m} \beta_k^* K^*(u_k^*, x^*). \tag{12}$$

[4] In this example generator $P(x^*, y|x)$ is intelligent since for any *picture* of the event x it describes the *essence* of the event. Using description of the essence of the event makes classification of the event an easy problem.

[5] In machine learning, they are called the reduced number of support vectors [2].

We then call the functions $K^*(u_k^*, x^*),\ k = 1, \ldots, m$ the *frames* (fragments) of knowledge. Our knowledge

$$f_\ell^*(x^*) = \sum_{k=1}^{m} \beta_k^* K^*(u_k^*, x^*) + b$$

is defined as a linear combination of the frames.

In the described terms, knowledge transfer from X^* into X requires the following:

1. To find the fundamental elements of knowledge $u_1^*, ..., u_m^*$ in space X^*.
2. To find frames (m functions) $K^*(u_1^*, x^*), ..., K^*(u_m^*, x^*)$ in space X^*.
3. To find the functions $\phi_1(x), \ldots, \phi_m(x)$ in space X such that

$$\phi_k(x_i) \approx K^*(u_k^*, x_i^*) \tag{13}$$

holds true for almost all pairs (x_i, x_i^*) generated by Intelligent Teacher that uses some (unknown) generator $P(x^*, y|x)$.

Note that the capacity of the set of functions from which $\phi_k(x)$ are to be chosen can be smaller than that of the capacity of the set of functions from which the classification function $y = f_\ell(x)$ is chosen (function $\phi_k(x)$ approximates just one fragment of knowledge, not the entire knowledge as function $y = f_\ell^*(x^*)$, which is a linear combination (12) of frames). Also, estimates of all the functions $\phi_1(x), ..., \phi_m(x)$ are done using different pairs as training sets of the same size ℓ. We hope that transfer of m fragments of knowledge from space X^* into space X can be done with higher accuracy than estimating function $y = f_\ell(x)$ from data (7).

After finding approximation of frames in space X, the knowledge about the rule obtained in space X^* can be approximated in space X as

$$f_\ell(x) \approx \sum_{k=1}^{m} \delta_k \phi_k(x) + b^*,$$

where coefficients $\delta_k = \alpha_k^*$ (taken from (10)) if approximations (13) are accurate. Otherwise, coefficients δ_k can be estimated from the training data.

More generally, in order to transfer knowledge from space X^* to space X one has to make the following two transformations in the training triplets (9):

1. Transform n-dimensional vectors of $x_i = (x_i^1, \ldots, x_i^n)^T$ into k-dimensional vectors $\mathcal{F}x_i = (\phi_1(x_i), \ldots, \phi_k(x_i))^T$. In order to transform vector x, one constructs m-dimensional space as follows: for any frame $K^*(x^*, x_s^*), s = 1, \ldots, k$ in space X^*, one construct it image (function) $\phi_s(x)$ in space X that defines the relationship

$$\phi_s(x) = \int K(x_s^*, x^*) P(x^*|x) dx^*, \qquad s = 1, \ldots, k.$$

This requires to solve the following regression estimation problem: given data

$$(x_1, z_1^s), \ldots, (x_\ell, z_\ell^s), \quad \text{where} \quad z_i^s = K(x_s^*, x_i^*),$$

find regression functions $\phi_s(x),\ s = 1, ..., k$, forming the space $\mathcal{F}(x) = (\phi_1(x), \ldots, \phi_k(x))^T$.

2. Use the target values s_i^* obtained for x_i^* in rule (10) instead of the values y_i given for x_i in triplet (9), i.e., replace target value y_i in triplets (9) with scores s_i^* given (10).

Thus the knowledge transfer algorithm transforms the training triplet[6]

$$((\mathcal{F}x_1, x_1^*, s_1^*), \ldots, (\mathcal{F}x_\ell, x_\ell^*, s_\ell^*)), \tag{14}$$

and then uses triplets (14) instead of triplets (9).

6 Feature-Based Algorithm for Knowledge Transfer

In this section, we present scalable algorithms of knowledge transfer in LUPI based on multivariate regressions of privileged features as functions of decision variables; we also illustrate the algorithms' performance and their properties on synthetic examples.

We assume again that we are given a set of iid triplets

$$(x_1, x_1^*, y_1), \ldots, (x_\ell, x_\ell^*, y_\ell), \quad x_i \in X = R^n, \quad x_i^* \in X^* = R^m, \quad y_i \in \{-1, +1\},$$

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 $X^* = R^m$.

In order to create knowledge transfer from space X^*, we use training data $x_1, \ldots, x_\ell$ to construct m multivariate regression functions $\phi_i(x^1, \ldots, x^n)$, where $i = 1, \ldots, m$, from n-dimensional decision space X to each of our m privileged features. Various types of regression could be used for that purpose, such as linear ridge regression or nonlinear kernel regression. After those regressions ϕ_i are constructed, we replace, for each $j = 1, \ldots, \ell$ and each $i = 1, \ldots, m$, the ith coordinate of jth privileged vector x_j^* with its regressed approximation $\phi_i(x_j^1, \ldots, x_j^n)$. In the next step, we construct the modified training dataset, consisting of m-dimensional regression-based replacements of privileged vectors. As a result, our modified training data will form the matrix

$$\begin{pmatrix} y_1 & \phi_1(x_1^1, \ldots, x_1^n) & \cdots & \phi_m(x_1^1, \ldots, x_1^n) \\ y_2 & \phi_1(x_2^1, \ldots, x_2^n) & \cdots & \phi_m(x_2^1, \ldots, x_2^n) \\ \cdots & \cdots & \cdots & \cdots \\ y_\ell & \phi_1(x_\ell^1, \ldots, x_\ell^n) & \cdots & \phi_m(x_\ell^1, \ldots, x_\ell^n) \end{pmatrix}.$$

Then, we apply some standard SVM algorithm to this modified training data and construct an m-dimensional decision rule. This rule can be used to classify any n-dimensional test vector $z = (z^1, \ldots, z^n)$ by executing the following steps:

[6] In the simplified version, pairs $(\mathcal{F}x_i, s_i^*),\ i = 1, \ldots, \ell$..

1. Using previously constructed (at the training stage) m multivariate regressions $\phi_1, \ldots, \phi_m$, compute m approximations to the unavailable privileged variables (coordinates) and form the m-dimensional vector

$$z^* = (\phi_1(z^1, \ldots, z^n), \phi_2(z^1, \ldots, z^n), \ldots, \phi_m(z^1, \ldots, z^n)).$$

2. Apply the constructed m-dimensional SVM decision rule to this m-dimensional augmented test vector z^*.

The described algorithm of knowledge transfer completely solves the main scalability problem of SVM+ algorithm, which was not practical for problems with more than several hundred training samples. Indeed, for larger number of samples, the SVM+ matrix for quadratic programming becomes ill-conditioned and larger number of parameters makes the problem of SVM+ parameter selection very time consuming [9]. In contrast to that, while the described knowledge transfer algorithm requires an additional step of calculating m multivariate regressions, which takes some limited time, this regression computation is performed only once during the whole process of parameter optimization (i.e., during grid search), and, most importantly, the augmented training data are then processed with any standard scalable SVM implementation.

In order to illustrate properties of the described knowledge transfer LUPI algorithm, consider its performance on the following simple synthetic example.

For training dataset, we generated ℓ two-dimensional random points (x^1, x^2), uniformly distributed in the square $[-1, +1] \times [-1, +1]$. Each point (x^1, x^2) was labeled with $y = \text{sgn}(x^1 + x^2)$. Both dimensions of these points were treated as standard decision features. In addition, for each point (x^1, x^2), we generated the value $x^3 = x^1 + x^2 + \varepsilon W$, where ε is the noise parameter, and W is an $N(0, 1)$-distributed random number; x^3 was treated as a privileged variable. Therefore, in this model, the privileged variable x^3 is more or less closely (depending on the noise level ε) related to the label of the decision vector (x^1, x^2).

We considered the following three types of classification scenarios:

- **SVM on decision features:** Training points (x^1, x^2) belong to the two-dimensional decision space, and RBF SVM is used to create the decision rule.
- **Knowledge transfer LUPI:** Training points (x^1, x^2) belong to the two-dimensional decision space, while privileged feature $(x_1^3, \ldots, x_\ell^3)^T$ belongs to the one-dimensional privileged space; knowledge transfer from privileged feature x^3 to the space of decision features $(x_1^1, \ldots, x_\ell^1)^T$ and $(x_1^2, \ldots, x_\ell^2)^T$ is realized with linear ridge regression. After augmenting x^1 and x^2 with regressed value of x^3, we construct the RBF SVM decision rule in the one-dimensional decision space.
- **SVM on privileged features:** Training points (x^3) belong to the one-dimensional decision space, and RBF SVM is used to create the decision rule.

For each of these scenarios, the error rate of the constructed decision rule was measured on the test dataset, generated according to the same distribution and containing (for statistical reliability of results) 10,000 two-dimensional

points (x^1, x^2). In our experiments, for each value of ℓ (selected as $10, 20, 40$) and each value of ε (selected as $0.01, 0.1, 1.0$), we generated 10 random realizations of training datasets of ℓ samples each. For each of these $10 \times 3 \times 3 = 90$ datasets, we ran all three classification scenarios (SVM on decision features, Knowledge trasnfer LUPI, and SVM on privileged features). Two parameters for RBF kernels (utilized in all three scenarios), namely SVM penalty parameter C and RBF kernel parameter γ, were selected using 6-fold cross-validation error rate over the two-dimensional grid of both parameters C and γ. In that grid, $\log_2(C)$ ranged of from -5 to $+5$ with step 0.5, and $\log_2(\gamma)$ ranged $+6$ to -6 with step 0.5 (thus the whole grid consisted of $21 \times 25 = 525$ pairs of tested parameters C and γ).

Table 1. Performance of SVMs and LUPI on synthetic example.

noise=0.01			
	training size 10	training size 20	training size 40
SVM on decision features	22.53 %	7.12 %	5.45 %
Knowledge transfer LUPI	10.10 %	2.32 %	1.77 %
SVM on privileged features	10.07 %	2.32 %	1.94 %
noise=0.1			
	training size 10	training size 20	training size 40
SVM on decision features	22.53 %	7.12 %	5.45 %
Knowledge transfer LUPI	10.22 %	2.30 %	2.06 %
SVM on privileged features	9.97 %	2.72 %	2.07 %
noise=1.0			
	training size 10	training size 20	training size 40
SVM on decision features	22.53 %	7.12 %	5.45 %
Knowledge transfer LUPI	18.24 %	5.74 %	3.44 %
SVM on privileged features	22.80 %	15.97 %	13.23 %

The averaged (over 10 realizations) error rates are shown in Table 1. The collected results suggest the following conclusions:

1. Knowledge Transfer LUPI improves the performance of Standard SVM on decision features (often significantly, in relative terms) in all of the considered scenarios. This relative improvement depends on interplay of noise and size of training sample.
2. For larger values of noise and/or larger training sizes, we observe that Knowledge Transfer LUPI can be even *better* that SVM on privileged features. While appearing counter-intuitive (an approximated (regressed) value turns out to be better for classification that the real one), this effect is due to the nature of synthetic distribution we used for this example. Indeed, for a large noise, the regressed privileged variable approximates the label function $\text{sgn}(x^1 + x^2)$

much more accurately (especially for large training size) than the actual data available during the training (since the accurate regression filters out most of the noise in the data). It also demonstrates the value of proper learning the structures of privileged space (with linear regression, in this example): if we learn these structures well, we might be able to improve performance significantly, even beyond the one delivered by SVM on privileged features.

Note that this is just one possible way to apply the idea of feature-based knowledge transfer. In many realistic examples, it is prudent not to switch completely from decision features to regressed privileged ones, but rather use both types of features in concatenation, thus forming the matrix of augmented training data

$$\begin{pmatrix} y_1 & x_1^1 \cdots x_1^n & \phi_1(x_1^1,\ldots,x_1^n) \cdots \phi_m(x_1^1,\ldots,x_1^n) \\ y_2 & x_2^1 \cdots x_2^n & \phi_1(x_2^1,\ldots,x_2^n) \cdots \phi_m(x_2^1,\ldots,x_2^n) \\ \cdots & \cdots & \cdots \cdots \cdots \\ y_\ell & x_\ell^1 \cdots x_\ell^n & \phi_1(x_\ell^1,\ldots,x_\ell^n) \cdots \phi_m(x_\ell^1,\ldots,x_\ell^n) \end{pmatrix}.$$

In this version of knowledge transfer LUPI, we apply some standard SVM algorithm to this augmented training data and construct an $(n+m)$-dimensional decision rule. This rule is then used to classify any test n-dimensional test vector $z = (z^1,\ldots,z^n)$ by executing the following steps:

1. Using previously constructed (at the training stage) m multivariate regressions $\phi_1,\ldots,\phi_m$, compute m approximations to the unavailable privileged variables (coordinates) and form the m-dimensional vector

$$z^* = (\phi_1(z^1,\ldots,z^n), \phi_2(z^1,\ldots,z^n),\ldots,\phi_m(z^1,\ldots,z^n)).$$

2. Concatenate the n-dimensional test vector z with this m-dimensional vector z^* to form augmented $(n+m)$-dimensional vector

$$(zz^*) = (z^1,\ldots,z^n,\phi_1(z^1,\ldots,z^n),\phi_2(z^1,\ldots,z^n),\ldots,\phi_m(z^1,\ldots,z^n))$$

3. Apply the $(n+m)$-dimensional SVM decision rule to this $(n+m)$-dimensional augmented test vector (zz^*).

In order to illustrate this version of knowledge transfer LUPI, we explored another synthetic dataset, derived from dataset "Parkinsons" in [8]. Since none of 22 features of "Parkinsons" dataset is privileged, we created several artificial scenarios emulating the presence of privileged information in that dataset. Specifically, we ordered "Parkinsons" features according to the values of their mutual information (with first features having the lowest mutual information, while the last features having the largest one). Then, for several values of parameter k, we treated the last k features as privileged ones, with the first $22-k$ features being treated as decision ones. Since our ordering was based on mutual information, these experiments corresponded to privileged spaces of various dimensions and various relevance levels for classification. For each considered value of k,

we generated 20 pairs of training and test subsets, containing, respectively 75 % and 25 % of elements of the "Parkinsons" dataset. For each of these pairs, we considered the following four types of classification scenarios:

- RBF SVM on $22 - k$ decision features;
- Knowledge transfer LUPI based on constructing k multivariate regressions from $22 - k$ decision features to each of k privileged ones, replacing the corresponding values in privileged vectors with their regressed approximations, and training RBF SVM on the augmented dataset consisting of 22 features;
- RBF SVM on k privileged features;
- RBF SVM on k all features.

In all these experiments, the parameters for RBF kernels were selected in the same way as for previous synthetic example.

Table 2. Performance of SVMs and LUPI on modified "Parkinsons" example.

k	SVM on decision features	Knowledge transfer LUPI	SVM on privileged features	SVM on all features
1	9.18 %	8.77 %	21.12 %	7.92 %
2	11.33 %	10.21 %	18.37 %	7.92 %
3	12.24 %	9.67 %	12.96 %	7.92 %
4	15.20 %	13.47 %	13.06 %	7.92 %
5	16.22 %	13.78 %	12.40 %	7.92 %
6	16.35 %	12.36 %	11.71 %	7.92 %
7	16.81 %	13.55 %	11.63 %	7.92 %
8	17.02 %	14.12 %	11.12 %	7.92 %
9	17.50 %	13.16 %	10.98 %	7.92 %
10	17.91 %	15.61 %	10.71 %	7.92 %

The averaged (over 20 realizations) error rates for these scenarios are shown in Table 2. The collected results suggest the following conclusions:

1. Knowledge Transfer LUPI improves the performance of Standard SVM on decision features (often significantly, in relative terms) in all of the considered scenarios. The error rates of LUPI are between SVMs constructed on decision features and on all features. In other words, if the error rate of SVM on decision features is B, while the error rate of SVM on all features is C, the error rate A of LUPI satisfies the bounds $C < A < B$. So one can evaluate the efficiency of LUPI approach by computing the metric $(B - A)/(B - C)$, which describes how much of the performance gap $B - C$ can be recovered by LUPI. In Table 2, this metric varies between 23 % and 59 %. Generally, in realistic examples, the typical value for this LUPI efficiency metric is in the ballpark of 35 %. Also note that if the gap $B - C$ is small compared to

C, it means that the privileged information is not particularly relevant; in that case, it is likely hopeless to apply LUPI anyway: there is little space for improvement for that. It is probably safe to start looking for LUPI solution if the gap $B - C$ is at least $1.5 - 2$ times larger than C.

2. The error rate of SVM on privileged features only becomes better than that of SVM on decision features for values of k larger than 3. This suggests that it is safer to rely on both decision and regressed privileged features in LUPI construction, since privileged features alone may not be sufficient to replace the classification information contained in decision features.

7 Conclusions

In this paper, we presented several properties of privileged information including its role in machine learning, its structure, and its applications. We extended the previous research in the area of privileged information by highlighting structures in the space of privileged information and various mechanisms that can leverage those structures for producing better solutions of pattern recognition problems. In particular, we presented a simple scalable algorithm for knowledge transfer, which avoids the scalability problem of current SVM+ implementations of LUPI. This algorithm is just a first step in the proposed direction, and its further improvements (especially concerning proper selection of relevant privileged features) will be the subject of future work.

References

1. Brachman, R., Levesque, H.: Knowledge Representation and Reasoning. Morgan Kaufmann, San Francisco (2004)
2. Burges, C.: Simplified support vector decision rules. In: 13th International Conference on Machine Learning, pp. 71–77 (1996)
3. Fouad, S., Tino, P., Raychaudhury, S., Schneider, P.: Incorporating privileged information through metric learning. IEEE Trans. Neural Netw. Learn. Syst. **24**, 1086–1098 (2013)
4. Ilin, R., Streltsov, S., Izmailov, R.: Learning with privileged information for improved target classification. Int. J. Monit. Surveill. Technol. Res. **2**(3), 5–66 (2014)
5. Izmailov, R., Vapnik, V., Vashist, A.: Multidimensional splines with infinite number of knots as SVM Kernels. In: International Joint Conference on Neural Networks, pp. 1096–1102. IEEE Press, New York (2013)
6. Kimeldorf, G., Wahba, G.: some results on Tchebycheffian spline functions. J. Math. Anal. Appl. **33**, 82–95 (1971)
7. Kolmogorov, A.: Mathematics as a Profession. Nauka, Moscow (1988). (in Russian)
8. Lichman, M.: UCI machine learning repository. University of California, School of Information and Computer Science, Irvine, CA (2013). http://archive.ics.uci.edu/ml
9. Pechyony, D., Izmailov, R., Vashist, A., Vapnik, V.: SMO-style algorithms for learning using privileged information. In: 2010 International Conference on Data Mining, pp. 235–241 (2010)

10. Ribeiro, B., Silva, C., Chen, N., Vieirac, A., das Nevesd, J.C.: Enhanced default risk models with SVM+. Expert Syst. Appl. **39**, 10140–10152 (2012)
11. Schölkopf, B., Herbrich, R., Smola, A.J.: A generalized representer theorem. In: Helmbold, D.P., Williamson, B. (eds.) COLT 2001 and EuroCOLT 2001. LNCS (LNAI), vol. 2111, pp. 416–426. Springer, Heidelberg (2001)
12. Sharmanska, V., Lampert, C.: Learning to rank using privileged information. In: 2013 IEEE International Conference on Computer Vision (ICCV), pp. 825–832 (2013)
13. Vapnik, V.: The Nature of Statistical Learning Theory. Springer-Verlag, New York (1995)
14. Vapnik, V.: Statistical Learning Theory. Wiley, New York (1998)
15. Vapnik, V.: Estimation of Dependencies Based on Empirical Data, 2nd edn. Springer, New York (2006)
16. Vapnik, V., Vashist, A.: A new learning paradigm: learning using privileged information. Neural Netw. **22**, 546–557 (2009)
17. Vapnik, V., Izmailov, R.: Learning with intelligent teacher: similarity control and knowledge transfer. In: Gammerman, A., Vovk, V., Papadopoulos, H. (eds.) SLDS 2015. LNCS, vol. 9047, pp. 3–32. Springer, Heidelberg (2015)
18. Vapnik, V., Izmailov, R.: Learning using privileged information: similarity control and knowledge transfer. J. Mach. Learn. Res. **16**, 2023–2049 (2015)
19. Yang, H., Patras, I.: 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)