LETTER Communicated by Léon Bottou

Gradient-Based Optimization of Hyperparameters

Yoshua Bengio
Département d'informatique et recherche opérationnelle, Université de Montréal, Montréal, Québec, Canada, H3C 3J7

Many machine learning algorithms can be formulated as the minimization of a training criterion that involves a hyperparameter. This hyperparameter is usually chosen by trial and error with a model selection criterion. In this article we present a methodology to optimize several hyperparameters, based on the computation of the gradient of a model selection criterion with respect to the hyperparameters. In the case of a quadratic training criterion, the gradient of the selection criterion with respect to the hyperparameters is efficiently computed by backpropagating through a Cholesky decomposition. In the more general case, we show that the implicit function theorem can be used to derive a formula for the hyperparameter gradient involving second derivatives of the training criterion.

1 Introduction

Machine learning algorithms pick a function from a set of functions $\mathcal{F}$ in order to minimize something that cannot be measured, only estimated, that is, the expected generalization performance of the chosen function. Many machine learning algorithms can be formulated as the minimization of a training criterion, which involves a hyperparameter, kept fixed during this minimization. For example, in the regularization framework (Tikhonov & Arsenin, 1977; Poggio, Torre, & Koch, 1985), one hyperparameter controls the strength of the penalty term: a larger penalty term reduces the "complexity" of the resulting function (it forces the solution to lie in a "smaller" subset of $\mathcal{F}$). A common example is *weight decay* (Hinton, 1987), used with neural networks and linear regression (also known in that case as ridge regression; Hoerl & Kennard, 1970), to penalize the L2-norm of the parameters. Increasing the penalty term (increasing the weight decay hyperparameter) corresponds to reducing the *effective capacity* (Guyon, Vapnik, Boser, Bottou, & Solla, 1992) by forcing the solution to be in a zero-centered hypersphere of smaller radius, which may improve generalization. A regularization term can also be interpreted as an a priori probability distribution on $\mathcal{F}$: in that case the weight decay is a scale parameter (e.g., inverse variance) of that distribution.

A model selection criterion can be used to select hyperparameters, more generally to compare and choose among models that may have a different capacity. Many model selection criteria have been proposed in the past

(Vapnik, 1982; Akaike, 1974; Craven & Wahba, 1979). When there is only a single hyperparameter, one can easily explore how its value affects the model selection criterion: typically one tries a finite number of values of the hyperparameter and picks the one that gives the lowest value of the model selection criterion.

In this article, we present a methodology to select simultaneously many hyperparameters using the gradient of the model selection criterion with respect to the hyperparameters. This methodology can be applied when some differentiability and continuity conditions of the training criterion are satisfied. The use of multiple hyperparameters has already been proposed in the Bayesian literature. One hyperparameter per input feature was used to control the prior on the parameters associated with that input feature (MacKay, 1995; Neal, 1998). In this case, the hyperparameters can be interpreted as scale parameters for the prior distribution on the parameters, for different directions in parameter space. Note that independently of this work, Larsen et al. (1998) have proposed a procedure similar to the one presented here, for optimizing regularization parameters of a neural network (different weight decays for different layers of the network). In sections 2, 3, and 4, we explain how the gradient with respect to the hyperparameters can be computed. In the conclusion, we briefly describe the results of preliminary experiments performed with the proposed methodology (described in more detail in Latendresse, 1999, and Bengio & Dugas, 1999), and we raise some important open questions concerning the kind of "overfitting" that can occur with the proposed methodology.

2 Objective Functions for Hyperparameters

We are given a set of independent data points, $D = \{z_1, \ldots, z_T\}$, all generated by the same unknown distribution $P(Z)$. We are given a set of functions $\mathcal{F}$ indexed by a parameter $\theta \in \Omega$ (i.e., each value of the parameter θ corresponds to a function in $\mathcal{F}$). In our applications we will have $\Omega \subseteq \mathcal{R}^s$. We would like to choose a value of θ that minimizes the expectation $E_Z(Q(\theta, Z))$ of a given loss functional $Q(\theta, Z)$. In supervised learning problems, we have input-output pairs $Z = (X, Y)$, with $X \in \mathcal{X}$, $Y \in \mathcal{Y}$, and θ is associated with a function f_θ from $\mathcal{X}$ to $\mathcal{Y}$. For example, we will consider the case of the quadratic loss, with real-valued vectors $\mathcal{Y} \subseteq \mathcal{R}^m$ and $Q(\theta, (X, Y)) = \frac{1}{2}(f_\theta(X) - Y)'(f_\theta(X) - Y)$. Note that we use the letter θ to represent parameters and the letter λ to represent hyperparameters.

In the next section, we will provide a formulation for the cases in which Q is quadratic in θ (e.g., quadratic loss with constant or affine function sets). In section 4, we will consider more general classes of functions and loss, which may be applied to the case of multilayer neural networks, for example.

2.1 Training Criteria. In its most general form, a training criterion C is any real-valued function of the set of empirical losses $Q(\theta, z_i)$ and of some

hyperparameters λ:

$$C(\theta, \lambda, D) = c(\lambda, Q(\theta, z_1), Q(\theta, z_2), \ldots, Q(\theta, z_T)).$$

Here λ is assumed to be a real-valued vector $\lambda = (\lambda_1, \ldots, \lambda_q)$. The proposed method relies on the assumption that C is continuous and differentiable almost everywhere with respect to θ and λ. When the hyperparameters are fixed, the learning algorithm attempts to perform the following minimization:

$$\theta(\lambda, D) = \text{argmin}_\theta \, C(\theta, \lambda, D). \tag{2.1}$$

An example of a training criterion with hyperparameters is

$$C = \sum_{(x_i, y_i) \in D} w_i(\lambda)(f_\theta(x_i) - y_i)^2 + \theta' A(\lambda)\theta, \tag{2.2}$$

where the hyperparameters provide different quadratic penalties to different parameters (with the matrix $A(\lambda)$), and different weights to different training patterns (with $w_i(\lambda)$), (as in Bengio & Dugas, 1999; Latendresse, 1999).

2.2 Model Selection Criteria. The model selection criterion E is used to select hyperparameters or more generally to choose one model among several. Ideally, it should be the expected generalization error (for a fixed λ), but $P(Z)$ is unknown, so many alternatives—approximations, bounds, or empirical estimates—have been proposed. Most model selection criteria have been proposed for selecting a single hyperparameter that controls the "complexity" of the class of functions in which the learning algorithm finds a solution, such as the minimum description length principle (Rissanen, 1990), structural risk minimization (Vapnik, 1982), the Akaike Information Criterion (Akaike, 1974), or the generalized cross-validation criterion (Craven & Wahba, 1979). Other criteria are those based on held-out data, such as the cross-validation estimates of generalization error. These are almost unbiased estimates of generalization error (Vapnik, 1982) obtained by testing f_θ on data not used to choose θ. For example, the K-fold cross-validation estimate uses K partitions of D, $S_1^1 \cup S_2^1, S_1^2 \cup S_2^2, \ldots, S_1^K \cup S_2^K$:

$$E_{cv}(\lambda, D) = \frac{1}{K} \sum_i \frac{1}{|S_2^i|} \sum_{z_t \in S_2^i} Q(\theta(\lambda, S_1^i), z_t).$$

When θ is fixed, the empirical risk $\frac{1}{T}\sum_t Q(\theta, z_t)$ is an unbiased estimate of the generalization error of f_θ (but it becomes an optimistic estimate when θ is chosen to minimize the empirical risk). Similarly, when λ is fixed, the cross-validation criterion is an almost unbiased estimate (when K approaches $|D|$) of the generalization error of $\theta(\lambda, D)$. When λ is chosen to minimize the

cross-validation criterion, this minimum value also becomes an optimistic estimate. And when there is a greater diversity of values $Q(f_{\theta(\lambda,D)}, z)$ that can be obtained for different values of λ, there is more risk of overfitting the hyperparameters. In this sense, the use of hyperparameters proposed in this article can be very different from the common use in which a hyperparameter helps to control overfitting. Instead, a blind use of the extra freedom brought by many hyperparameters could deteriorate generalization.

3 Optimizing Hyperparameters for a Quadratic Training Criterion

In this section we analyze the simpler case in which the training criterion C is a quadratic polynomial of the parameters θ. The dependence on the hyperparameters λ can be of higher order, as long as it is continuous and differentiable almost everywhere (see, for example, Bottou, 1998, for more detailed technical conditions sufficient for stochastic gradient descent):

$$C = a(\lambda) + b(\lambda)'\theta + \frac{1}{2}\theta' H(\lambda)\theta, \tag{3.1}$$

where $\theta, b \in \mathcal{R}^s$, $a \in \mathcal{R}$, and $H \in \mathcal{R}^{s\times s}$. For a minimum of equation 3.1 to exist requires that H be positive definite. It can be obtained by solving the linear system

$$\frac{\partial C}{\partial \theta} = b + H\theta = 0, \tag{3.2}$$

which yields the solution

$$\theta(\lambda) = -H^{-1}(\lambda)b(\lambda). \tag{3.3}$$

Assuming that E depends only on λ through θ, the gradient of the model selection criterion E with respect to λ is

$$\frac{\partial E}{\partial \lambda} = \frac{\partial E}{\partial \theta}\frac{\partial \theta}{\partial \lambda}.$$

If there were a direct dependency (not through θ), an extra partial derivative would have to be added. For example, in the case of the cross-validation criteria,

$$\frac{\partial E_{cv}}{\partial \theta} = \frac{1}{K}\sum_i \frac{1}{|S_2^i|}\sum_{z_t \in S_2^i} \frac{\partial Q(\theta, z_t)}{\partial \theta}.$$

In the quadratic case, the influence of λ on θ is spelled out by equation 3.3, yielding

$$\frac{\partial \theta_i}{\partial \lambda} = -\sum_j \frac{\partial H^{-1}_{i,j}}{\partial \lambda} b_j - \sum_j H^{-1}_{i,j}\frac{\partial b_j}{\partial \lambda}. \tag{3.4}$$

Although the second sum can be readily computed, $\frac{\partial H^{-1}_{i,j}}{\partial \lambda}$ in the first sum is more challenging; we consider several methods below. One solution is based on the computation of gradients through the inverse of a matrix. This general but inefficient solution is the following:

$$\frac{\partial H^{-1}_{i,j}}{\partial \lambda} = \sum_{k,l} \frac{\partial H^{-1}_{i,j}}{\partial H_{k,l}} \frac{\partial H_{k,l}}{\partial \lambda},$$

where

$$\frac{\partial H^{-1}_{i,j}}{\partial H_{k,l}} = -H^{-1}_{i,j} H^{-1}_{l,k} + I_{i \neq l, j \neq k} H^{-1}_{i,j} \, minor(H, j, i)^{-1}_{l',k'}, \tag{3.5}$$

where $minor(H, j, i)$ denotes the "minor matrix," obtained by removing the jth row and the ith column from H, and the indices (l', k') in equation 3.5 refer to the position within a minor matrix that corresponds to the position (l, k) in H (note $l \neq i$ and $k \neq j$). Unfortunately, the computation of this gradient requires $O(s^5)$ multiply-add operations for an $s \times s$ matrix H, which is much more than is required by the inversion of H ($O(s^3)$ operations). A better solution is based on the following equality: $HH^{-1} = I$, where I is the $s \times s$ identity matrix. This implies, by differentiating with respect to λ: $\frac{\partial H}{\partial \lambda} H^{-1} + H \frac{\partial H^{-1}}{\partial \lambda} = 0$. Isolating $\frac{\partial H^{-1}}{\partial \lambda}$, we get

$$\frac{\partial H^{-1}}{\partial \lambda} = -H^{-1} \frac{\partial H}{\partial \lambda} H^{-1}, \tag{3.6}$$

which requires only about $2s^3$ multiply-add operations.

An even better solution (which was suggested by Léon Bottou) is to return to equation 3.2, which can be solved using about $s^3/3$ multiply-add operations (when $\theta \in \mathcal{R}^s$). The idea is to backpropagate gradients through each of the operations performed to solve the linear system. The objective is to compute the gradient of E with respect to H and b through the effect of H and b on θ, in order to compute $\frac{\partial E}{\partial \lambda}$, as illustrated in Figure 1. The backpropagation costs the same as the linear system solution—about $s^3/3$ multiply/add operations—so this is the approach that we have kept for our implementation. Since H is the Hessian matrix, it is positive definite and symmetric, and equation 3.2 can be solved through the Cholesky decomposition of H (assuming H is full rank, which is likely if the hyperparameters provide some sort of weight decay). The Cholesky decomposition of a symmetric positive definite matrix H gives $H = LL'$ where L is a lower diagonal matrix (with zeros above the diagonal). It is computed in time $O(s^3)$ as follows:

$$\begin{aligned}
&\texttt{for } i = 1, \ldots, s \\
&\qquad L_{i,i} = \sqrt{H_{i,i} - \textstyle\sum_{k=1}^{i-1} L^2_{i,k}} \\
&\qquad \text{for } j = i + 1, \ldots, s \\
&\qquad\qquad L_{j,i} = \left(H_{i,j} - \textstyle\sum_{k=1}^{i-1} L_{i,k} L_{j,k} \right) / L_{i,i}.
\end{aligned}$$

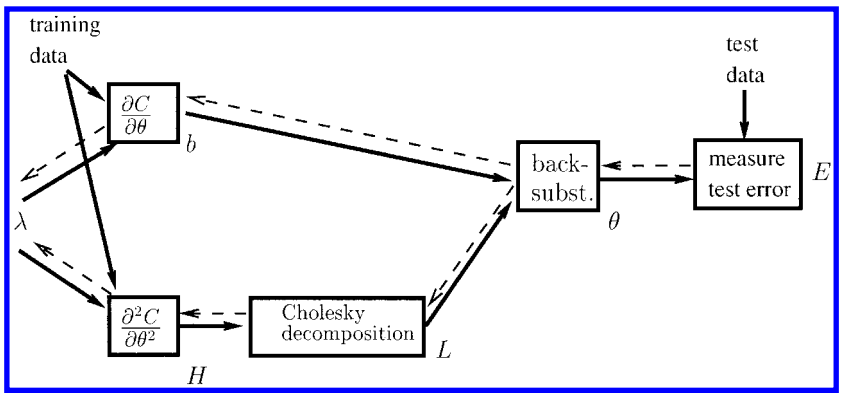

Figure 1: Illustration of forward paths (full lines) and gradient paths (dashed) for computation of the model selection criterion E and its derivative with respect to the hyperparameters (λ), when using the method based on the Cholesky decomposition and backsubstitution to solve for the parameters (θ).

Once the Cholesky decomposition is achieved, the linear system $LL'\theta = -b$ can be easily solved in two backsubstitution steps: (1) first solve $Lu = -b$ for u; (2) solve $L'\theta = u$ for θ:

1. Iterate once forward through the rows of L:

$$\texttt{for } i = 1, \ldots, s,$$
$$u_i = (-b_i - \textstyle\sum_{k=1}^{i-1} L_{i,k} u_k)/L_{i,i}.$$

2. Iterate once backward through the rows of L:

$$\texttt{for } i = s, \ldots, 1,$$
$$\theta_i = (u_i - \textstyle\sum_{k=i+1}^{s} L_{k,i} \theta_k)/L_{i,i}.$$

The computation of the gradient of θ with respect to the elements of H and b proceeds in exactly the reverse order. We start by backpropagating through the backsubstitution steps, and then through the Cholesky decomposition. Together, the three algorithms that follow allow computing $\frac{\partial E}{\partial b}$ and $\frac{\partial E}{\partial H}$, starting from the "partial" parameter gradient $\left.\frac{\partial E}{\partial \theta_i}\right|_{\theta_1,\ldots,\theta_s}$ (not taking into account the dependencies of θ_i on θ_j for $j > i$). As intermediate results, the algorithm computes the partial derivatives with respect to u and L, as well as the "full gradients" with respect to θ, $\left.\frac{\partial E}{\partial \theta_i}\right|_{\theta_i}$, taking into account all the dependencies between the θ_i's brought by the recursive computation of the θ_i's.

First backpropagate through the solution of $L'\theta = u$:

$$\texttt{initialize dEdtheta} \leftarrow \left.\frac{\partial E}{\partial \theta}\right|_{\theta_1,\ldots,\theta_s}$$
$$\texttt{initialize dEdL} \leftarrow 0$$

```
for i = 1, ..., s
        dEdu_i ← dEdtheta_i / L_{i,i}
        dEdL_{i,i} ← dEdL_{i,i} − dEdtheta_i θ_i / L_{i,i}
        for k = i + 1 ... s
                dEdtheta_k ← dEdtheta_k − dEdtheta_i L_{k,i} / L_{i,i}
                dEdL_{k,i} ← dEdL_{k,i} − dEdtheta_i θ_k / L_{i,i}.
```

Then backpropagate through the solution of $Lu = -b$:

```
for i = s, ..., 1
        ∂E/∂b_i ← −dEdu_i / L_{i,i}
        dEdL_{i,i} ← dEdL_{i,i} − dEdu_i u_i / L_{i,i}
        for k = 1, ..., i − 1
                dEdu_k ← dEdu_k − dEdu_i L_{i,k} / L_{i,i}
                dEdL_{i,k} ← dEdL_{i,k} − dEdu_i u_k / L_{i,i}.
```

The above algorithm gives the gradient of the model selection criterion E with respect to coefficient $b(\lambda)$ of the training criterion, as well as with respect to the lower diagonal matrix L, $\mathtt{dEdL}_{i,j} = \frac{\partial E}{\partial L_{i,j}}$.

Finally, we backpropagate through the Cholesky decomposition, to convert the gradients with respect to L into gradients with respect to the Hessian $H(\lambda)$:

```
for i = s, ..., 1
        for j = s, ..., i + 1
                dEdL_{i,i} ← dEdL_{i,i} − dEdL_{j,i} L_{j,i} / L_{i,i}
                ∂E/∂H_{i,j} ← dEdL_{j,i} / L_{i,i}
                for k = 1, ..., i − 1
                        dEdL_{i,k} ← dEdL_{i,k} − dEdL_{j,i} L_{j,k} / L_{i,i}
                        dEdL_{j,k} ← dEdL_{j,k} − dEdL_{j,i} L_{i,k} / L_{i,i}
        ∂E/∂H_{i,i} ← (1/2) dEdL_{i,i} / L_{i,i}
        for k = 1, ..., i − 1
                dEdL_{i,k} ← dEdL_{i,k} − dEdL_{i,i} L_{i,k} / L_{i,i}.
```

Note that we have computed gradients only with respect to the diagonal and upper diagonal of H because H is symmetric. Once we have the gradients of E with respect to b and H, we use the functional form of $b(\lambda)$ and $H(\lambda)$ to compute the gradient of E with respect to λ:

$$\frac{\partial E}{\partial \lambda} = \sum_i \frac{\partial E}{\partial b_i}\frac{\partial b_i}{\partial \lambda} + \sum_{i,j} \frac{\partial E}{\partial H_{i,j}}\frac{\partial H_{i,j}}{\partial \lambda}$$

(again we assumed that there is no direct dependency from λ to E; otherwise, an extra term must be added).

Using this approach rather than the one described in the previous subsection, the overall computation of gradients is therefore in about $s^3/3$ multiply-

add operations rather than $O(s^5)$. The most expensive step is the backpropagation through the Cholesky decomposition itself (three nested s-iterations loops). This step may be shared if there are several linear systems to solve with the same Hessian matrix. For example, this will occur in linear regression with multiple outputs because H is block-diagonal, with one block for each set of parameters associated to one output and all blocks being the same (equal to the input "design matrix" $\sum_t x_t x_t'$, denoting x_t the input training vectors). Only a single Cholesky computation needs to be done, shared across all blocks.

3.1 Weight Decays for Linear Regression. In this section, we illustrate the method in the particular case of multiple weight decays for linear regression, with K-fold cross-validation as the model selection criterion. The hyperparameter λ_j will be a weight decay associated with the jth input variable. The training criterion for the kth partition is

$$C_k = \frac{1}{|S_1^k|} \sum_{(x_t, y_t) \in S_1^k} \frac{1}{2}(\Theta x_t - y_t)'(\Theta x_t - y_t) + \frac{1}{2} \sum_j \lambda_j \sum_i \Theta_{i,j}^2. \tag{3.7}$$

The objective is to penalize separately each of the input variables (as in MacKay, 1995; Neal, 1998), a kind of "soft variable selection" (see Latendresse, 1999, for more discussion and experiments with this setup). The training criterion is quadratic, as in equation 3.1, with coefficients

$$a = \frac{1}{2} \sum_t y_t' y_t, \qquad b_{(ij)} = -\sum_t y_{t,i} x_{t,j},$$
$$H_{(ij),(i'j')} = \delta_{i,i'} \sum_t x_{t,j} x_{t,j'} + \delta_{i,i'} \delta_{j,j'} \lambda_j,$$

where $\delta_{i,j} = 1$ when $i = j$ and 0 otherwise, and (ij) is an index corresponding to indices (i, j) in the weight matrix Θ, for example, $(ij) = (i-1) \times s + j$. From the above definition of the coefficients of C, we obtain their partial derivatives with respect to λ:

$$\frac{\partial b}{\partial \lambda} = 0, \qquad \frac{\partial H_{(ij),(i'j')}}{\partial \lambda_k} = \delta_{i,i'} \delta_{j,j'} \delta_{j,k}.$$

By plugging the above definitions of the coefficients and their derivatives in the equations and algorithms of the previous section, we have therefore obtained an algorithm for computing the gradient of the model selection criterion with respect to the input weight decays of a linear regression.

Note that here H is block-diagonal, with m identical blocks of size $(n+1)$, so the Cholesky decomposition (and similarly backpropagating through it) can be performed in about $(s/m)^3/3$ multiply-add operations rather than $s^3/3$ operations, where m is the number of outputs (the dimension of the output variable).

4 Nonquadratic Criterion: Hyperparameters Gradient

If the training criterion C is not quadratic in terms of the parameters θ, it will in general be necessary to apply an iterative numerical optimization algorithm to minimize the training criterion. In this section we consider what happens after this minimization is performed—at a value of θ where $\partial C/\partial\theta$ is approximately zero and $\partial^2 C/\partial\theta^2$ is positive definite (otherwise we would not be at a minimum of C). The minimization of $C(\theta, \lambda, D)$ defines a function $\theta(\lambda, D)$ (see equation 2.1). With our assumption of smoothness of C, the implicit function theorem tells us that this function exists locally and is differentiable. To obtain this function, we write

$$F(\theta, \lambda) = \frac{\partial C}{\partial \theta} = 0,$$

evaluated at $\theta = \theta(\lambda, D)$ (at a minimum of C). Differentiating the above equation with respect to λ, we obtain

$$\frac{\partial F}{\partial \theta}\frac{\partial \theta}{\partial \lambda} + \frac{\partial F}{\partial \lambda} = 0,$$

so we obtain a general formula for the gradient of the fitted parameters with respect to the hyperparameters:

$$\frac{\partial \theta(\lambda, D)}{\partial \lambda} = -\left(\frac{\partial^2 C}{\partial \theta^2}\right)^{-1} \frac{\partial^2 C}{\partial \lambda \partial \theta} = -H^{-1}\frac{\partial^2 C}{\partial \lambda \partial \theta}. \tag{4.1}$$

Let us see how this result relates to the special case of a quadratic training criterion, $C = a + b'\theta + \frac{1}{2}\theta' H\theta$:

$$\frac{\partial \theta}{\partial \lambda} = -H^{-1}\left(\frac{\partial b}{\partial \lambda} + \frac{\partial H}{\partial \lambda}\theta\right) = -H^{-1}\frac{\partial b}{\partial \lambda} + H^{-1}\frac{\partial H}{\partial \lambda}H^{-1}b,$$

where we have substituted $\theta = -H^{-1}b$. Using the equality 3.6, we obtain the same formula as in equation 3.4.

Let us consider more closely the case of a neural network with one layer of hidden units with hyperbolic tangent activations, a linear output layer, squared loss, and hidden-layer weights $W_{i,j}$. For example, if we want to use hyperparameters for penalizing the use of inputs, we have a criterion similar to equation 3.7:

$$C_k = \frac{1}{|S_1^k|}\sum_{(x_t, y_t)\in S_1^k} \frac{1}{2}(f_\theta(x_t) - y_t)'(f_\theta(x_t) - y_t) + \frac{1}{2}\sum_j \lambda_j \sum_i W_{i,j}^2,$$

with $C = \sum_k C_k$, and the cross-derivatives are easy to compute:

$$\frac{\partial^2 C}{\partial W_{i,j}\partial \lambda_k} = \delta_{k,j} W_{i,j}.$$

The Hessian and its inverse require more work, but can be done respectively in at most $O(s^2)$ and $O(s^3)$ operations. See, for example, Bishop (1992) for the exact computation of the Hessian for multilayer neural networks. See Becker and LeCun (1989) and LeCun, Denker, and Solla (1990) for a diagonal approximation, which can be computed and inverted in $O(s)$ operations.

5 Summary of Experiments and Conclusions

We have presented a new methodology for simultaneously optimizing several hyperparameters, based on the computation of the gradient of a model selection criterion with respect to the hyperparameters, taking into account the influence of the hyperparameters on the parameters. We have considered the simpler case of a training criterion that is quadratic with respect to the parameters ($\theta \in \mathcal{R}^s$) and the more general nonquadratic case. We have shown a particularly efficient procedure in the quadratic case that is based on backpropagating gradients through the Cholesky decomposition and backsubstitutions. This was an improvement: we have arrived at this $s^3/3$-operations procedure after studying first an $O(s^5)$ procedure and then a $2s^3$-operations procedure. In the case of input weight decays for linear regression, the computation can even be reduced to about $(s/m)^3/3$ operations when there are m outputs.

We have performed preliminary experiments with the proposed methodology in several simple cases, using conjugate gradients to optimize the hyperparameters. The application to linear regression with weight decays for each input is described in Latendresse (1999). The hyperparameter optimization algorithm is used to perform a soft selection of the input variables. A large weight decay on one of the inputs effectively forces the corresponding weights to very small values. Comparisons on simulated data sets are made in Latendresse (1999) with ordinary regression as well as with stepwise regression methods and the adaptive ridge (Grandvalet, 1998) or LASSO (Tibshirani, 1995).

Another type of application of the proposed method has been explored, in the context of a real-world problem of nonstationary time-series prediction (Bengio & Dugas, 1999). In this case, an extension of the cross-validation criterion to sequential data that may be nonstationary is used. Because of this nonstationarity, recent data may sometimes be more relevant to current predictions than older data. The training criterion is a sum of weighted errors for the past examples, and these weights are given by a parameterized function of time (as the $w_i(\lambda)$ in equation 2.2). The parameters of that function are two hyperparameters that control when a transition in the unknown generating process would have occurred and how strong that change was or should be trusted. In these experiments, the weight given to past data points is a sigmoid function of the time: the threshold and the slope of the sigmoid are the hyperparameters, representing, respectively, the time of a strong transition and the strength of that transition. Optimizing

these hyperparameters, we obtained statistically significant improvements in predicting one-month-ahead future volatility of Canadian stocks. The comparisons were made against several linear, constant, and ARMA models of the volatility. The experiments were performed on monthly return data from 473 Canadian stocks from 1976 to 1996. The measure of performance is the average out-of-sample squared error in predicting the squared returns. Single-sided significance tests were performed, taking into account the auto-covariance in the temporal series of errors and the covariance of the errors between the compared models. When comparing the prediction of the first moment (expected return), no model significantly improved on the historical average of stock returns (constant model). When comparing the prediction of the second moment (expected squared returns), the method based on hyperparameters optimization beats all the other methods, with a p-value of 1% or less.

What remains to be done? First, we need more experiments, in particular with the nonquadratic case (e.g., MLPs), and with model selection criteria other than cross-validation (which has large variance; Breiman, 1996). Second, there are important theoretical questions that remain unanswered concerning the amount of overfitting that can be brought when too many hyperparameters are optimized. As we outlined in the introduction, the situation with hyperparameters may be compared with the situation of parameters. However, whereas the form of the training criterion as a sum of independent errors allows defining the capacity for a class of functions and relating it to the difference between generalization error and training error, more work needs to be done to obtain a similar analysis for hyperparameters.

Acknowledgments

The author would like to thank Léon Bottou, Pascal Vincent, François Blanchette, and François Gingras, as well as the NSERC Canadian funding agency.

References

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control, AC-19*, 716–728.

Becker, S., & LeCun, Y. (1989). Improving the convergence of back-propagation learning with second order methods. In D. Touretzky, G. Hinton, & T. Sejnowski (Eds.), *Proceedings of the 1988 Connectionist Models Summer School* (pp. 29–37). San Mateo, CA: Morgan Kaufmann.

Bengio, Y., & Dugas, C. (1999). *Learning simple non-stationarities with hyperparameters.* Submitted.

Bishop, C. (1992). Exact calculation of the Hessian matrix for the multi-layer perceptron. *Neural Computation, 4*, 494–501.

Bottou, L. (1998). Online algorithms and stochastic approximations. In D. Saad (Ed.), *Online learning in neural networks*. Cambridge: Cambridge University Press.

Breiman, L. (1996). Heuristics of instability and stabilization in model selection. *Annals of Statistics, 24*, 2350–2383.

Craven, P., & Wahba, G. (1979). Smoothing noisy data with spline functions. *Numerical Mathematics, 31*, 377–403.

Grandvalet, Y. (1998). Least absolute shrinkage is equivalent to quadratic penalization. In L. Niklasson, M. Boden, & T. Ziemske (Eds.), *ICANN'98* (pp. 201–206). Berlin: Springer-Verlag.

Guyon, I., Vapnik, V., Boser, B., Bottou, L., & Solla, S. A. (1992). Structural risk minimization for character recognition. In J. Moody, S. Hanson, & R. Lipmann (Eds.), *Advances in neural information processing systems, 4* (pp. 471–479). San Mateo, CA: Morgan Kaufmann.

Hinton, G. (1987). Learning translation invariant in massively parallel networks. In J. de Bakker, A. Nijman, & P. Treleaven (Eds.), *Proceedings of PARLE Conference on Parallel Architectures and Languages Europe* (pp. 1–13). Berlin: Springer-Verlag.

Hoerl, A., & Kennard, R. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics, 12*, 55–67.

Larsen, J., Svarer, C., Andersen, L. N., & Hansen, L. K. (1998). Adaptive regularization in neural networks modeling. In G. B. Orr & K.-R. Muller (Eds.), *Neural Networks: Tricks of the Trade* (pp. 113–132). Berlin: Springer-Verlag.

Latendresse, S. (1999). *Utilisation d'hyper-parametres pour la selection de variable*. M.Sc. thesis, University of Montreal.

LeCun, Y., Denker, J., & Solla, S. (1990). Optimal brain damage. In D. Touretzky (Ed.), *Advances in neural information processing systems, 2* (pp. 598–605). San Mateo, CA: Morgan Kaufmann.

MacKay D. (1995). Probable networks and plausible predictions—a review of practical Bayesian methods for supervised neural networks. *Network: Computation in Neural Systems, 6*, 469–505. Available online at: http://www.iro.umontreal.ca/∼lisa.

Neal, R. (1998). Assessing relevance determination methods using delve. In C. Bishop (Ed.), *Neural networks and machine learning* (pp. 97–129). Berlin: Springer-Verlag.

Poggio, T., Torre, V., & Koch, C. (1985). Computational vision and regularization theory. *Nature, 317*, 314–319.

Rissanen, J. (1990). *Stochastic complexity in statistical inquiry*. Singapore: World Scientific.

Tibshirani, R. (1995). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society B, 58*, 267–288.

Tikhonov, A., & Arsenin, V. (1977). *Solutions of ill-posed problems*. Washington, DC: W. H. Winston.

Vapnik, V. (1982). *Estimation of dependences based on empirical data*. Berlin: Springer-Verlag.

Received June 1, 1999; accepted August 30, 1999.

This article has been cited by:

1. Zefang Shen, R. A. Viscarra Rossel. 2021. Automated spectroscopic modelling with optimised convolutional neural networks. *Scientific Reports* **11**:1. . [Crossref]
2. Thao Nguyen, Devin C. Francom, D.J. Luscher, J.W. Wilkerson. 2021. Bayesian calibration of a physics-based crystal plasticity and damage model. *Journal of the Mechanics and Physics of Solids* **149**, 104284. [Crossref]
3. Muhammed Maruf Öztürk. 2021. A tuned feed-forward deep neural network algorithm for effort estimation. *Journal of Experimental & Theoretical Artificial Intelligence* **345**, 1-25. [Crossref]
4. Haimiao Zhang, Baodong Liu, Hengyong Yu, Bin Dong. 2021. MetaInv-Net: Meta Inversion Network for Sparse View CT Image Reconstruction. *IEEE Transactions on Medical Imaging* **40**:2, 621-634. [Crossref]
5. Sean Kauffman, Murray Dunne, Giovani Gracioli, Waleed Khan, Nirmal Benann, Sebastian Fischmeister. 2021. Palisade: A framework for anomaly detection in embedded systems. *Journal of Systems Architecture* **113**, 101876. [Crossref]
6. Alexander Y. Sun, Bridget R. Scanlon, Himanshu Save, Ashraf Rateb. 2021. Reconstruction of GRACE Total Water Storage Through Automated Machine Learning. *Water Resources Research* **57**:2. . [Crossref]
7. Xin He, Kaiyong Zhao, Xiaowen Chu. 2021. AutoML: A survey of the state-of-the-art. *Knowledge-Based Systems* **212**, 106622. [Crossref]
8. Yingze Tian, Baoguo Wu, Xiaohui Su, Yan Qi, Yuling Chen, Zhiqiang Min. 2021. A Crown Contour Envelope Model of Chinese Fir Based on Random Forest and Mathematical Modeling. *Forests* **12**:1, 48. [Crossref]
9. Nandana Sengupta, Fallaw Sowell. 2020. On the Asymptotic Distribution of Ridge Regression Estimators Using Training and Test Samples. *Econometrics* **8**:4, 39. [Crossref]
10. Min-gon Chu, Baehyun Min, Seoyoon Kwon, Gayoung Park, Sungil Kim, Nguyen Xuan Huy. 2020. Determination of an infill well placement using a data-driven multi-modal convolutional neural network. *Journal of Petroleum Science and Engineering* **195**, 106805. [Crossref]
11. Yoram Bachrach, Richard Everett, Edward Hughes, Angeliki Lazaridou, Joel Z. Leibo, Marc Lanctot, Michael Johanson, Wojciech M. Czarnecki, Thore Graepel. 2020. Negotiating team formation using deep reinforcement learning. *Artificial Intelligence* **288**, 103356. [Crossref]
12. Li Yang, Abdallah Shami. 2020. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing* **415**, 295-316. [Crossref]
13. Guang Li, Ren Togo, Takahiro Ogawa, Miki Haseyama. Soft-Label Anonymous Gastric X-Ray Image Distillation 305-309. [Crossref]
14. Yanan Guo, Xiaoqun Cao, Bainian Liu, Mei Gao. 2020. Solving Partial Differential Equations Using Deep Learning and Physical Constraints. *Applied Sciences* **10**:17, 5917. [Crossref]
15. Carlos Perales-González, Mariano Carbonero-Ruz, Javier Pérez-Rodríguez, David Becerra-Alonso, Francisco Fernández-Navarro. 2020. Negative correlation learning in the extreme learning machine framework. *Neural Computing and Applications* **32**:17, 13805-13823. [Crossref]
16. William W. Tso, Baris Burnak, Efstratios N. Pistikopoulos. 2020. HY-POP: Hyperparameter optimization of machine learning models through parametric programming. *Computers & Chemical Engineering* **139**, 106902. [Crossref]
17. Lerina Aversano, Mario Luca Bernardi, Marta Cimitile, Riccardo Pecori. Fuzzy Neural Networks to Detect Parkinson Disease 1-8. [Crossref]
18. Lerina Aversano, Mario Luca Bernardi, Marta Cimitile, Riccardo Pecori. Early Detection of Parkinson Disease using Deep Neural Networks on Gait Dynamics 1-8. [Crossref]
19. Hyejung Chung, Kyung-shik Shin. 2020. Genetic algorithm-optimized multi-channel convolutional neural network for stock market prediction. *Neural Computing and Applications* **32**:12, 7897-7914. [Crossref]
20. Abderrachid Hamrani, Abdolhamid Akbarzadeh, Chandra A. Madramootoo. 2020. Machine learning for predicting greenhouse gas emissions from agricultural soils. *Science of The Total Environment* 140338. [Crossref]
21. Anshul Mittal, Swati Aggarwal. 2020. Hyperparameter Optimization Using Sustainable Proof of Work in Blockchain. *Frontiers in Blockchain* **3**. . [Crossref]
22. Shane T. Barratt, Stephen P. Boyd. 2020. Least squares auto-tuning. *Engineering Optimization* **1**, 1-22. [Crossref]
23. Hongying Liu, Ruyi Luo, Fanhua Shang, Xuechun Meng, Shuiping Gou, Biao Hou. 2020. Semi-Supervised Deep Metric Learning Networks for Classification of Polarimetric SAR Data. *Remote Sensing* **12**:10, 1593. [Crossref]

24. Yi-Qi Hu, Yang Yu. 2020. A technical view on neural architecture search. *International Journal of Machine Learning and Cybernetics* **11**:4, 795-811. [Crossref]
25. Mohammad Loni, Sima Sinaei, Ali Zoljodi, Masoud Daneshtalab, Mikael Sjödin. 2020. DeepMaker: A multi-objective optimization framework for deep neural networks in embedded systems. *Microprocessors and Microsystems* **73**, 102989. [Crossref]
26. Davy Preuveneers, Ilias Tsingenopoulos, Wouter Joosen. 2020. Resource Usage and Performance Trade-offs for Machine Learning Models in Smart Environments. *Sensors* **20**:4, 1176. [Crossref]
27. Liesle Caballero, Mario Jojoa, Winston S. Percybrooks. 2020. Optimized neural networks in industrial data analysis. *SN Applied Sciences* **2**:2. . [Crossref]
28. V. I. Avrutskiy. Preventing Overfitting by Training Derivatives 144-163. [Crossref]
29. Maria Tsiakmaki, Georgios Kostopoulos, Sotiris Kotsiantis, Omiros Ragos. 2020. Implementing AutoML in Educational Data Mining for Prediction Tasks. *Applied Sciences* **10**:1, 90. [Crossref]
30. Jorge G. Madrid, Hugo Jair Escalante, Eduardo Morales. Meta-learning of Textual Representations 57-67. [Crossref]
31. Renlong Jie, Junbin Gao, Andrey Vasnev, Minh-Ngoc Tran. 2020. HyperTube: A Framework for Population-Based Online Hyperparameter Optimization with Resource Constraints. *IEEE Access* **8**, 69038-69057. [Crossref]
32. Guoqiang Zhong, Wencong Jiao, Wei Gao, Kaizhu Huang. 2020. Automatic Design of Deep Networks with Neural Blocks. *Cognitive Computation* **12**:1, 1-12. [Crossref]
33. Hiram Ponce, Paulo Souza. A Comparative Analysis of Evolutionary Learning in Artificial Hydrocarbon Networks 223-234. [Crossref]
34. Benlin Liu, Yongming Rao, Jiwen Lu, Jie Zhou, Cho-Jui Hsieh. MetaDistiller: Network Self-Boosting via Meta-Learned Top-Down Distillation 694-709. [Crossref]
35. Yaoyao Liu, Bernt Schiele, Qianru Sun. An Ensemble of Epoch-Wise Empirical Bayes for Few-Shot Learning 404-421. [Crossref]
36. Nicoletta Del Buono, Flavia Esposito, Laura Selicato. Methods for Hyperparameters Optimization in Learning Approaches: An Overview 100-112. [Crossref]
37. Hongying Liu, Tianwen Zhu, Fanhua Shang, Yuanyuan Liu, Derui Lv, Yang Shuyuan. 2020. Deep Fuzzy Graph Convolutional Networks for PolSAR Imagery Pixel-wise Classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 1-1. [Crossref]
38. Dounia El Bourakadi, Ali Yahyaouy, Jaouad Boumhidi. 2019. Multi-Agent System Based on the Extreme Learning Machine and Fuzzy Control for Intelligent Energy Management in Microgrid. *Journal of Intelligent Systems* **29**:1, 877-893. [Crossref]
39. Vishal Passricha, Rajesh Kumar Aggarwal. 2019. PSO-based optimized CNN for Hindi ASR. *International Journal of Speech Technology* **22**:4, 1123-1133. [Crossref]
40. Laurent Parmentier, Olivier Nicol, Laetitia Jourdan, Marie-Eleonore Kessaci. TPOT-SH: A Faster Optimization Algorithm to Solve the AutoML Problem on Large Datasets 471-478. [Crossref]
41. Rohitash Chandra, Konark Jain, Ratneel V. Deo, Sally Cripps. 2019. Langevin-gradient parallel tempering for Bayesian neural learning. *Neurocomputing* **359**, 315-326. [Crossref]
42. YoungJun Yoo. 2019. Hyperparameter optimization of deep neural network using univariate dynamic encoding algorithm for searches. *Knowledge-Based Systems* **178**, 74-83. [Crossref]
43. Pierpaolo Croce, Filippo Zappasodi, Laura Marzetti, Arcangelo Merla, Vittorio Pizzella, Antonio Maria Chiarelli. 2019. Deep Convolutional Neural Networks for Feature-Less Automatic Classification of Independent Components in Multi-Channel Electrophysiological Brain Recordings. *IEEE Transactions on Biomedical Engineering* **66**:8, 2372-2380. [Crossref]
44. Patrick Schratz, Jannes Muenchow, Eugenia Iturritxa, Jakob Richter, Alexander Brenning. 2019. Hyperparameter tuning and performance assessment of statistical and machine-learning algorithms using spatial data. *Ecological Modelling* **406**, 109-120. [Crossref]
45. Minsu Cho, Chinmay Hegde. Reducing the Search Space for Hyperparameter Optimization Using Group Sparsity 3627-3631. [Crossref]
46. Krishanu Maity, Satyabrata Maity, Nimisha Ghosh. A Bi-Level Approach for Hyper-Parameter Tuning of an Evolutionary Extreme Learning Machine 124-129. [Crossref]
47. C. Soize, C. Farhat. 2019. Probabilistic learning for modeling and quantifying model-form uncertainties in nonlinear computational mechanics. *International Journal for Numerical Methods in Engineering* **117**:7, 819-843. [Crossref]

48. Noemí DeCastro-García, Ángel Luis Muñoz Castañeda, David Escudero García, Miguel V. Carriegos. 2019. Effect of the Sampling of a Dataset in the Hyperparameter Optimization Phase over the Efficiency of a Machine Learning Algorithm. *Complexity* **2019**, 1-16. [Crossref]
49. Lars Kotthoff, Chris Thornton, Holger H. Hoos, Frank Hutter, Kevin Leyton-Brown. Auto-WEKA: Automatic Model Selection and Hyperparameter Optimization in WEKA 81-95. [Crossref]
50. Manuel Eugenio Morocho-Cayamcela, Haeyoung Lee, Wansu Lim. 2019. Machine Learning for 5G/B5G Mobile and Wireless Communications: Potential, Limitations, and Future Directions. *IEEE Access* **7**, 137184-137206. [Crossref]
51. Ang Li, Ola Spyra, Sagi Perel, Valentin Dalibard, Max Jaderberg, Chenjie Gu, David Budden, Tim Harley, Pramod Gupta. A Generalized Framework for Population Based Training 1791-1799. [Crossref]
52. Raji Ghawi, Jürgen Pfeffer. 2019. Efficient Hyperparameter Tuning with Grid Search for Text Categorization using kNN Approach with BM25 Similarity. *Open Computer Science* **9**:1, 160-180. [Crossref]
53. Rémi Flamary, Marco Cuturi, Nicolas Courty, Alain Rakotomamonjy. 2018. Wasserstein discriminant analysis. *Machine Learning* **107**:12, 1923-1945. [Crossref]
54. Hesameddin Mohammadi, Meisam Razaviyayn, Mihailo R. Jovanovic. Variance Amplification of Accelerated First-Order Algorithms for Strongly Convex Quadratic Optimization Problems 5753-5758. [Crossref]
55. Hongyuan Zhan, Gabriel Gomes, Xiaoye S. Li, Kamesh Madduri, Kesheng Wu. Efficient Online Hyperparameter Learning for Traffic Flow Prediction 164-169. [Crossref]
56. C. Soize. 2018. Design optimization under uncertainties of a mesoscale implant in biological tissues using a probabilistic learning algorithm. *Computational Mechanics* **62**:3, 477-497. [Crossref]
57. Antonio Maria Chiarelli, Pierpaolo Croce, Arcangelo Merla, Filippo Zappasodi. 2018. Deep learning for hybrid EEG-fNIRS brain–computer interface: application to motor imagery classification. *Journal of Neural Engineering* **15**:3, 036028. [Crossref]
58. Jean Feng, Noah Simon. 2018. Gradient-based Regularization Parameter Selection for Problems With Nonsmooth Penalty Functions. *Journal of Computational and Graphical Statistics* **27**:2, 426-435. [Crossref]
59. Christopher J. Fariss, Zachary M. Jones. 2018. Enhancing Validity in Observational Settings When Replication is Not Possible. *Political Science Research and Methods* **6**:2, 365-380. [Crossref]
60. Simon Jenni, Paolo Favaro. Deep Bilevel Learning 632-648. [Crossref]
61. Qianyu Zhang, Bin Li, Yi Wu. Evolutionary Structure Optimization of Convolutional Neural Networks for Deployment on Resource Limited Systems 742-753. [Crossref]
62. Luca Oneto, Emanuele Fumeo, Giorgio Clerico, Renzo Canepa, Federico Papa, Carlo Dambra, Nadia Mazzino, Davide Anguita. 2017. Dynamic Delay Predictions for Large-Scale Railway Networks: Deep and Shallow Extreme Learning Machines Tuned via Thresholdout. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **47**:10, 2754-2767. [Crossref]
63. Roozbeh Razavi-Far, Mehrdad Saif, Vasile Palade, Enrico Zio. Adaptive incremental ensemble of extreme learning machines for fault diagnosis in induction motors 1615-1622. [Crossref]
64. JinXing Che, YouLong Yang, Li Li, YanYing Li, SuLing Zhu. 2017. A modified support vector regression: Integrated selection of training subset and model. *Applied Soft Computing* **53**, 308-322. [Crossref]
65. Pablo Ribalta Lorenzo, Jakub Nalepa, Luciano Sanchez Ramos, José Ranilla Pastor. Hyper-parameter selection in deep neural networks using parallel particle swarm optimization 1864-1871. [Crossref]
66. Masanori Suganuma, Shinichi Shirakawa, Tomoharu Nagao. A genetic programming approach to designing convolutional neural network architectures 497-504. [Crossref]
67. Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C. Lupu, Fabio Roli. Towards Poisoning of Deep Learning Algorithms with Back-gradient Optimization 27-38. [Crossref]
68. Gang Luo. 2016. A review of automatic selection methods for machine learning algorithms and hyper-parameter values. *Network Modeling Analysis in Health Informatics and Bioinformatics* **5**:1. . [Crossref]
69. Ole-Christoffer Granmo. Bayesian Unification of Gradient and Bandit-Based Learning for Accelerated Global Optimisation 222-226. [Crossref]
70. YouLong Yang, JinXing Che, YanYing Li, YanJun Zhao, SuLing Zhu. 2016. An incremental electric load forecasting model based on support vector regression. *Energy* **113**, 796-808. [Crossref]
71. Ronay Ak, Olga Fink, Enrico Zio. 2016. Two Machine Learning Approaches for Short-Term Wind Speed Time-Series Prediction. *IEEE Transactions on Neural Networks and Learning Systems* **27**:8, 1734-1747. [Crossref]

72. David K. Tcheng, Ashwin K. Nayak, Charless C. Fowlkes, Surangi W. Punyasena. 2016. Visual Recognition Software for Binary Classification and Its Application to Spruce Pollen Identification. *PLOS ONE* **11**:2, e0148879. [Crossref]
73. Joon-Hyung Kim, Joo-Hyun Rho. 2016. Numerical Study to Improve the Flow Uniformity of Blow-Down HVAC Duct System for a Train. *The KSFM Journal of Fluid Machinery* **19**:1, 18-23. [Crossref]
74. Manuel Martin Salvador, Marcin Budka, Bogdan Gabrys. Towards Automatic Composition of Multicomponent Predictive Systems 27-39. [Crossref]
75. Ermal Toto, Elke A. Rundensteiner, Yanhua Li, Richard Jordan, Mariya Ishutkina, Kajal Claypool, Jun Luo, Fan Zhang. PULSE: A Real Time System for Crowd Flow Prediction at Metropolitan Subway Stations 112-128. [Crossref]
76. Joel Ribeiro, Josep Carmona. A Method for Assessing Parameter Impact on Control-Flow Discovery Algorithms 181-202. [Crossref]
77. Adi Makmal, Alexey A. Melnikov, Vedran Dunjko, Hans J. Briegel. 2016. Meta-learning within Projective Simulation. *IEEE Access* **4**, 2110-2122. [Crossref]
78. Gang Luo. 2015. MLBCD: a machine learning tool for big clinical data. *Health Information Science and Systems* **3**:1. . [Crossref]
79. Frank Hutter, Jörg Lücke, Lars Schmidt-Thieme. 2015. Beyond Manual Tuning of Hyperparameters. *KI - Künstliche Intelligenz* **29**:4, 329-337. [Crossref]
80. Philip Erickson, Michael Cline, Nishith Tirpankar, Tom Henderson. Gaussian processes for multi-sensor environmental monitoring 208-213. [Crossref]
81. Olga Fink, Enrico Zio, Ulrich Weidmann. 2015. Fuzzy Classification With Restricted Boltzman Machines and Echo-State Networks for Predicting Potential Railway Door System Failures. *IEEE Transactions on Reliability* **64**:3, 861-868. [Crossref]
82. Olga Fink, Enrico Zio, Ulrich Weidmann. 2015. A Classification Framework for Predicting Components' Remaining Useful Life Based on Discrete-Event Diagnostic Data. *IEEE Transactions on Reliability* **64**:3, 1049-1056. [Crossref]
83. Francesco Ciucci, Chi Chen. 2015. Analysis of Electrochemical Impedance Spectroscopy Data Using the Distribution of Relaxation Times: A Bayesian and Hierarchical Bayesian Approach. *Electrochimica Acta* **167**, 439-454. [Crossref]
84. Fabian Bürger, Josef Pauli. A Holistic Classification Optimization Framework with Feature Selection, Preprocessing, Manifold Learning and Classifiers 52-68. [Crossref]
85. Olga Fink, Enrico Zio, Ulrich Weidmann. 2014. Quantifying the reliability of fault classifiers. *Information Sciences* **266**, 65-74. [Crossref]
86. Gavin C. Cawley, Nicola L.C. Talbot. 2014. Kernel learning at the first level of inference. *Neural Networks* **53**, 69-80. [Crossref]
87. Christopher J. Fariss, Zachary M. Jones. 2014. Enhancing Validity in Observational Settings When Replication is Not Possible. *SSRN Electronic Journal* . [Crossref]
88. Olga Fink, Enrico Zio, Ulrich Weidmann. 2013. Predicting time series of railway speed restrictions with time-dependent machine learning techniques. *Expert Systems with Applications* **40**:15, 6033-6040. [Crossref]
89. O Fink, U Weidmann, E Zio. Extreme learning machines for predicting operation disruption events in railway systems 1781-1787. [Crossref]
90. Balázs Kégl. 2013. Introduction to multivariate discrimination. *EPJ Web of Conferences* **55**, 02001. [Crossref]
91. Olga Fink, Andrew Nash, Ulrich Weidmann. 2013. Predicting Potential Railway Operational Disruptions with Echo State Networks. *Transportation Research Record: Journal of the Transportation Research Board* **2374**:1, 66-72. [Crossref]
92. E.A. Zanaty. 2012. Support Vector Machines (SVMs) versus Multilayer Perception (MLP) in data classification. *Egyptian Informatics Journal* **13**:3, 177-183. [Crossref]
93. Miguel Lázaro-Gredilla, Vanessa Gómez-Verdejo, Emilio Parrado-Hernández. 2012. Low-cost model selection for SVMs using local features. *Engineering Applications of Artificial Intelligence* **25**:6, 1203-1211. [Crossref]
94. Shijun Wang, Ronald M. Summers. 2012. Machine learning and radiology. *Medical Image Analysis* **16**:5, 933-951. [Crossref]
95. Yingqun Xiao, Lianggui Feng. 2012. A novel linear ridgelet network approach for analog fault diagnosis using wavelet-based fractal analysis and kernel PCA as preprocessors. *Measurement* **45**:3, 297-310. [Crossref]
96. Yingqun Xiao, Lianggui Feng. 2012. A novel neural-network approach of analog fault diagnosis based on kernel discriminant analysis and particle swarm optimization. *Applied Soft Computing* **12**:2, 904-920. [Crossref]
97. Yuh-Jye Lee, Yi-Ren Yeh, Hsing-Kuo Pao. Introduction to Support Vector Machines and Their Applications in Bankruptcy Prognosis 731-761. [Crossref]
98. Hong-xia Pang, Wen-de Dong, Zhi-hai Xu, Hua-jun Feng, Qi Li, Yue-ting Chen. 2011. Novel linear search for support vector machine parameter selection. *Journal of Zhejiang University SCIENCE C* **12**:11, 885-896. [Crossref]

99. Chaofan Dai, Yanghe Feng, Jianmai Shi. 2011. Evolutioinary Combination of models in DSS based on Genetic Programming. *Journal of Software* **6**:3. . [Crossref]
100. Yingqun Xiao, Yigang He. 2011. A novel approach for analog fault diagnosis based on neural networks and improved kernel PCA. *Neurocomputing* **74**:7, 1102-1115. [Crossref]
101. Alessandro Perolini. A Fast Approximated Evolutionary Approach to Improve SVM Accuracy 193-206. [Crossref]
102. A. W. Witoelar, A. Ghosh, J. J. G. de Vries, B. Hammer, M. Biehl. 2010. Window-Based Example Selection in Learning Vector Quantization. *Neural Computation* **22**:11, 2924-2961. [Abstract] [Full Text] [PDF] [PDF Plus]
103. R. Pavón, F. Díaz, R. Laza, M.V. Luzón. 2010. Experimental evaluation of an automatic parameter setting system. *Expert Systems with Applications* **37**:7, 5224-5238. [Crossref]
104. Kai Krajsek, Hanno Scharr. Diffusion filtering without parameter tuning: Models and inference tools 2536-2543. [Crossref]
105. Clément Chatelain, Sébastien Adam, Yves Lecourtier, Laurent Heutte, Thierry Paquet. 2010. A multi-model selection framework for unknown and/or evolutive misclassification cost problems. *Pattern Recognition* **43**:3, 815-823. [Crossref]
106. CLEMENT CHATELAIN, SEBASTIEN ADAM, YVES LECOURTIER, LAURENT HEUTTE, THIERRY PAQUET. 2010. NONCOST SENSITIVE SVM TRAINING USING MULTIPLE MODEL SELECTION. *Journal of Circuits, Systems and Computers* **19**:01, 231-242. [Crossref]
107. Ashwin Srinivasan, Ganesh Ramakrishnan. Parameter Screening and Optimisation for ILP Using Designed Experiments 217-225. [Crossref]
108. Mathias M. Adankon, Mohamed Cheriet. 2009. Model selection for the LS-SVM. Application to handwriting recognition. *Pattern Recognition* **42**:12, 3264-3270. [Crossref]
109. Ramesh Natarajan, Vikas Sindhwani, Shirish Tatikonda. Sparse Least-Squares Methods in the Parallel Machine Learning (PML) Framework 314-319. [Crossref]
110. Kegan G. G. Samuel, Marshall F. Tappen. Learning optimized MAP estimates in continuously-valued MRF models 477-484. [Crossref]
111. Reyes Pavón, Fernando Díaz, Rosalía Laza, Victoria Luzón. 2009. Automatic parameter tuning with a Bayesian case-based reasoning system. A case of study. *Expert Systems with Applications* **36**:2, 3407-3420. [Crossref]
112. Reyes Pavón, Fernando Díaz, Victoria Luzón. 2008. A model for parameter setting based on Bayesian networks. *Engineering Applications of Artificial Intelligence* **21**:1, 14-25. [Crossref]
113. Hua-chao YANG, Shu-bi ZHANG, Ka-zhong DENG, Pei-jun DU. 2007. Research into a Feature Selection Method for Hyperspectral Imagery Using PSO and SVM. *Journal of China University of Mining and Technology* **17**:4, 473-478. [Crossref]
114. Sen Jia, Yuntao Qian. 2007. Spectral and Spatial Complexity-Based Hyperspectral Unmixing. *IEEE Transactions on Geoscience and Remote Sensing* **45**:12, 3867-3879. [Crossref]
115. Chien-Ming Huang, Yuh-Jye Lee, Dennis K.J. Lin, Su-Yun Huang. 2007. Model selection for support vector machines via uniform design. *Computational Statistics & Data Analysis* **52**:1, 335-346. [Crossref]
116. Behrooz Makki, Nasser Sadati, Mona Noori Hosseini. Modeling Superconductive Fault Current Limiter Using Constructive Neural Networks 2859-2863. [Crossref]
117. B. Makki, S.A. Seyedsalehi, N. Sadati, M. Noori Hosseini. Voice conversion using nonlinear principal component analysis 336-339. [Crossref]
118. A. Rakotomamonjy. 2007. Analysis of SVM regression bounds for variable ranking. *Neurocomputing* **70**:7-9, 1489-1501. [Crossref]
119. Mathias M. Adankon, Mohamed Cheriet. 2007. Optimizing resources in model selection for support vector machine. *Pattern Recognition* **40**:3, 953-963. [Crossref]
120. Liefeng Bo, Ling Wang, Licheng Jiao. 2006. Feature Scaling for Kernel Fisher Discriminant Analysis Using Leave-One-Out Cross Validation. *Neural Computation* **18**:4, 961-978. [Abstract] [PDF] [PDF Plus]
121. M.M. Adankon, M. Cheriet. New Formulation of SVM for Model Selection 1900-1907. [Crossref]
122. Sambu Seo, K. Obermayer. Dynamic Hyperparameter Scaling Method for LVQ Algorithms 3196-3203. [Crossref]
123. Abhijit Kulkarni, V.K. Jayaraman, B.D. Kulkarni. 2004. Support vector classification with parameter tuning assisted by agent-based technique. *Computers & Chemical Engineering* **28**:3, 311-318. [Crossref]
124. A. Passerini, M. Pontil, P. Frasconi. 2004. New Results on Error Correcting Output Codes of Kernel Machines. *IEEE Transactions on Neural Networks* **15**:1, 45-54. [Crossref]
125. Giovanna Castellano, Anna Maria Fanelli, Corrado Mencar. 2002. A neuro-fuzzy network to generate human-understandable knowledge from data. *Cognitive Systems Research* **3**:2, 125-144. [Crossref]

126. Nedjem-Eddine Ayat, Mohamed Cheriet, Ching Y. Suen. Optimization of the SVM Kernels Using an Empirical Error Minimization Scheme 354-369. [Crossref]
127. Yi Wu, Ching-King Lin, E.Y. Chang, J.R. Smith. Multimodal information fusion for video concept detection 2391-2394. [Crossref]
128. N. Chapados, Y. Bengio. Input decay: simple and effective soft variable selection 1233-1237. [Crossref]
129. M.M. Adankon, M. Cheriet, N.F. Ayat. Optimizing resources in model selection for support vector machines 925-930. [Crossref]
130. Baofu Duan, Yoh-Han Pao. Iterative feature weighting for identification of relevant features with radial basis function networks 1063-1068. [Crossref]