K-Nearest Neighbors Directed Noise Injection in Multilayer Perceptron Training

M. Skurichina, Š. Raudys, and R. P. W. Duin

***Abstract*—The relation between classifier complexity and learning set size is very important in discriminant analysis. One of the ways to overcome the complexity control problem is to add noise to the training objects, increasing in this way the size of the training set. Both the amount and the directions of noise injection are important factors which determine the effectiveness for classifier training. In this paper the effect is studied of the injection of Gaussian spherical noise and k-nearest neighbors directed noise on the performance of multilayer perceptrons. As it is impossible to provide an analytical investigation for multilayer perceptrons, a theoretical analysis is made for statistical classifiers. The goal is to get a better understanding of the effect of noise injection on the accuracy of sample-based classifiers. By both empirical as well as theoretical studies, it is shown that the k-nearest neighbors directed noise injection is preferable over the Gaussian spherical noise injection for data with low intrinsic dimensionality.**

***Index Terms*—Intrinsic data dimensionality, k-nearest neighbors directed noise injection, multilayer perceptrons, noise injection, Parzen window classifier.**

I. Introduction

In DISCRIMINANT analysis one often has to face the *small training sample size problem*. This arises when the data feature space dimensionality is large compared with the number of available training objects. Sometimes large difficulties appear in constructing a discriminant function on small training sample sets, resulting in discriminant functions having a bad performance [1], [2]. In order to make a good choice for the classification rule or to judge the training sample size it is important to be familiar with the small sample properties of the sample-based classification rules. These properties can be characterized by the difference and/or by the ratio of the generalization error and the asymptotic probability of misclassification.

Small sample properties of statistical classifiers depend on their complexity and on the data dimensionality [10], [12], [18]. For a data model with multivariate Gaussian distributed pattern classes having a common covariance matrix with r significant nonzero eigenvalues, it was shown that the generalization errors of the nearest mean classifier [11], the Parzen window classifier [13], and the zero empirical error classifier [14] depend on a true (intrinsic) data dimensionality r. Our simulation experiments have also confirmed that for this data model the small sample properties of the nonlinear single layer perceptron are not determined by the dimensionality of the feature space p but by the intrinsic dimensionality r.

A well-known technique to solve the small training sample size problem involves the generation of more training objects by *noise injection* (NI) to the training data [3], [9]. Usually, spherical Gaussian distributed noise is generated around each training object. In case of high-dimensional data, however, it may happen that the intrinsic data dimensionality is smaller than the dimensionality of the feature space, in which the data are represented. The data are thereby located in a subspace of the feature space. Moreover, different measurements (features) may have large differences in scale due to their different nature. In this case, besides the variance of the added noise, also the directions of the noise become important. When spherical noise is injected to the data with low intrinsic dimensionality, noise is also added in the directions where no data are located. By this, spherical NI can distort the distribution of the training sample set and destroy the low intrinsic dimensionality of the data. Consequently, the small sample properties of the classifier will be deteriorated. The injection of Gaussian noise in only the direction of the k-nearest neighbors of an object may be more effective, as the local distribution of the training set is taken into account. This noise will be called the *k-nearest neighbors* (k-NN) directed noise, which basic idea is given in [5]. The k-NN abbreviation should not be confused with the k-NN classifier which is not studied or used in this paper.

In this paper we study the effectiveness of *Gaussian noise injection* (GNI) and *k-NN directed noise injection* (k-NN DNI) on the classifier performance. We also study the effect of the intrinsic data dimensionality on the small sample size properties of classifiers and on the effectiveness of NI. Our theoretical study and simulations show that, both, the direction and the variance of noise are very important in NI.

It is shown theoretically, in Section II, for the case of statistical classification rules, why the k-NN DNI may be more preferable than Gaussian NI in the case of low intrinsic dimensionalities. For the case of multilayer perceptrons a theoretical analysis involves nonlinearities and leads to a very complex algebra. Therefore we perform a simulation study on the effectiveness of NI in perceptron training. The results of this simulation study are presented in Section III. Conclusions are summarized in Section IV.

II. Noise Injection and Statistical Classifiers

Let us consider a noise injection model, in which R independent p-variate random Gaussian noise vectors $\boldsymbol{Z}_{jr}^{(i)} \sim N(\boldsymbol{0}, \lambda^2 \boldsymbol{I})$ $(i = 1, 2;\ j = 1, 2, \cdots, M;\ r = 1, 2, \cdots, R)$ are added to each of M p-variate training vectors $\boldsymbol{X}_j^{(i)}$ from two pattern classes π_1 and π_2. As a result, $2MR$ "noisy" training vectors $\boldsymbol{U}_{jr}^{(i)} = \boldsymbol{X}_j^{(i)} + \boldsymbol{Z}_{jr}^{(i)}$ are obtained.

Manuscript received June 25, 1998; revised March 5, 1999 and December 20, 1999. This work was supported by the Foundation for Applied Sciences (STW) and the Dutch Organization for Scientific Research (NWO).

M. Skurichina and R. P. W. Duin are with the Pattern Recognition Group, Department of Applied Physics, Delft University of Technology, 2600GA Delft, The Netherlands (e-mail: {marina; duin}@ph.tn.tudelft.nl).

Š. Raudys is with the Department of Data Analysis, Institute of Mathematics and Informatics, Vilnius 2600, Lithuania (e-mail: sarunas.raudys@ktl.mii.lt).

Publisher Item Identifier S 1045-9227(00)03002-2.

In order to understand the noise effect upon an accuracy of sample-based classification rules we consider a couple of statistical classification rules where the investigation can be performed analytically. At first, in Sections II-A and II-B on the examples of the *Fisher linear discriminant* (FLD) [22] and the *Parzen window* (PW) classifier [23], we compare classifiers with built-in "analytical noise" and classifiers using NI. This shows the importance of the number of generated noise vectors on the classification performance. Under certain conditions the histogram classifier [16] using NI is similar to the PW density estimate. In Section II-C we compare these two classifiers in order to analyze the influence of the intrinsic data dimensionality on the effectiveness of NI. The latter study gives the theoretical basis for possible benefits of k-NN DNI compared with Gaussian NI for the data with low intrinsic dimensionality.

A. Linear Discriminant Analysis

The standard Fisher linear discriminant function [8], [22] is defined as

$$\hat{g}_{\mathrm{f}}(\boldsymbol{x}) = \boldsymbol{x}'\hat{\boldsymbol{w}}^{\mathrm{F}} + w_0 \tag{1}$$

with $\hat{\boldsymbol{w}}^{\mathrm{F}} = \boldsymbol{S}^{-1}(\overline{\boldsymbol{X}}^{(1)} - \overline{\boldsymbol{X}}^{(2)})$ and $w_0 = -(1/2)(\overline{\boldsymbol{X}}^{(1)} + \overline{\boldsymbol{X}}^{(2)})'\hat{\boldsymbol{w}}^{\mathrm{F}}$, where

$$\overline{\boldsymbol{X}}^{(i)} = \frac{1}{M}\sum_{j=1}^{M}\boldsymbol{X}_j^{(i)} \qquad (i = 1, 2)$$

and

$$\boldsymbol{S} = \frac{1}{2M-2}\sum_{i=1}^{2}\sum_{j=1}^{M}\left(\boldsymbol{X}_j^{(i)} - \overline{\boldsymbol{X}}^{(i)}\right)\left(\boldsymbol{X}_j^{(i)} - \overline{\boldsymbol{X}}^{(i)}\right)'$$

are the sample estimates of the ith class mean and the common covariance matrix, respectively.

Considering the noise injection model described above, we can design the FLD from $2MR$ "noisy" training vectors $\boldsymbol{U}_{jr}^{(i)}$ with the sample estimate of the ith class mean

$$\overline{\boldsymbol{U}}^{(i)} = \frac{1}{MR}\sum_{j=1}^{M}\sum_{r=1}^{R}\boldsymbol{U}_{jr}^{(i)} \qquad (i = 1, 2)$$

and obtain the following estimate of the covariance matrix:

$$\begin{aligned}\boldsymbol{S}^* &= \frac{1}{2M-2}\sum_{i=1}^{2}\sum_{j=1}^{M}\left(\boldsymbol{U}_{jr}^{(i)} - \overline{\boldsymbol{U}}^{(i)}\right)\left(\boldsymbol{U}_{jr}^{(i)} - \overline{\boldsymbol{U}}^{(i)}\right)' \\ &= \boldsymbol{S} + \boldsymbol{S}_M\end{aligned}$$

where

$$\boldsymbol{S}_M = \frac{1}{2MR-2}\sum_{i=1}^{2}\sum_{j=1}^{M}\sum_{r=1}^{R}\left(\boldsymbol{Z}_{jr}^{(i)} - \overline{\boldsymbol{Z}}_j^{(i)}\right)\left(\boldsymbol{Z}_{jr}^{(i)} - \overline{\boldsymbol{Z}}_j^{(i)}\right)'$$

is an additional random term which arises due to the random nature of noise vectors $\boldsymbol{Z}_{jr}^{(i)}$. Here $\overline{\boldsymbol{Z}}_j^{(i)} = (1/R)\sum_{r=1}^{R}\boldsymbol{Z}_{jr}^{(i)}$ is the mean of noise vectors generated around each training vector $\boldsymbol{X}_j^{(i)}$. Asymptotically, as the number of noise vectors tends to infinity $R \to \infty$, the sample estimate of the covariance matrix of noise vectors $\boldsymbol{Z}_{jr}^{(i)}$ tends to the diagonal matrix $\boldsymbol{S}_M \to \lambda^2\boldsymbol{I}$. Therefore we have

$$\begin{aligned}\boldsymbol{S}^* &\to \frac{1}{2M-2}\sum_{i=1}^{2}\sum_{j=1}^{M}\left(\boldsymbol{X}_j^{(i)} - \overline{\boldsymbol{X}}^{(i)}\right)\left(\boldsymbol{X}_j^{(i)} - \overline{\boldsymbol{X}}^{(i)}\right)' + \lambda^2\boldsymbol{I} \\ &= \boldsymbol{S} + \lambda^2\boldsymbol{I}\end{aligned}$$

which is used in the regularized discriminant function (RDF) [9], [24]. However, $\boldsymbol{S}_M \neq \lambda^2\boldsymbol{I}$ for finite R. Consequently, when R is finite, we have another classification rule, different from the RDF. Obviously, for small R a random nature of the matrix $\boldsymbol{S}_M$ will deteriorate the estimate $\boldsymbol{S}^*$ and increase the generalization error.

In the analysis of *multilayer perceptrons* (MLP's) it is important to notice that adding a supplementary weight decay term $\lambda^2\boldsymbol{w}'\boldsymbol{w}$ [21] to the cost function of the linear *single layer perceptron* (SLP) is equivalent to the RDF [4], [15] (in regression similar results were obtained in [20] and [25]). This means that asymptotically NI in the linear SLP design is equivalent to weight decay regularization and/or to the RDF. For finite R the matrix $\boldsymbol{S}_M$ is random, and, when R is small, most probably a worse classifier is obtained. Although this equivalence holds only for the linear SLP case, our simulation study [4] has shown that for the nonlinear SLP and MLP weight decay and NI perform similarly as for the linear SLP. Our simulation experiments [4] have also confirmed that NI in the case of finite R is less effective than weight decay—"analytical noise," and that with an increase in R the effectiveness of "analytical noise" and NI to the training vectors becomes similar.

B. Parzen Window Classifier

The standard version of the nonparametric *Parzen window* (PW) classifier [8], [23] is based on sample estimates of the class conditional densities of the following form:

$$\hat{f}_{\mathrm{PW}}(\boldsymbol{x}|\pi_i) = \frac{1}{M}\sum_{j=1}^{M}N\left(\boldsymbol{x}, \boldsymbol{X}_j^{(i)}, \lambda^2\boldsymbol{I}\right) \tag{2}$$

where

$$\begin{aligned}&N\left(\boldsymbol{x}, \boldsymbol{X}_j^{(i)}, \lambda^2\boldsymbol{I}\right) \\ &\quad= \frac{1}{\left(\sqrt{2\pi}\right)^p\lambda^p}\exp\left(-\frac{1}{2\lambda^2}\left(\boldsymbol{x} - \boldsymbol{X}_j^{(i)}\right)'\left(\boldsymbol{x} - \boldsymbol{X}_j^{(i)}\right)\right),\end{aligned}$$

λ^2 is called a smoothing parameter, and p is the dimensionality of the feature space Ω.

At a fixed point $\boldsymbol{x}$ the value of the PW density estimate $\hat{f}_{\mathrm{PW}}(\boldsymbol{x}|\pi_i)$ depends on M random training vectors $\boldsymbol{X}_1^{(i)}, \boldsymbol{X}_2^{(i)}, \cdots, \boldsymbol{X}_M^{(i)}$. Considering all possible training sets consisting of M observations, this density can be analyzed as a random variable. According to the central limit theorem the sum (2) of M random contribution terms $N(\boldsymbol{x}, \boldsymbol{X}_j^{(i)}, \lambda^2\boldsymbol{I})$ tends to the Gaussian distribution, when the training sample size $M \to \infty$. Thus, the conditional probability of misclassification at one particular point $\boldsymbol{x}$ can be approximated by the means E and by

the variances V of the sample estimates of the class conditional densities $\hat{f}_{\mathrm{PW}}(\boldsymbol{x}|\pi_i)$, $i = 1, 2$

$$P(\text{misclassification}|\boldsymbol{x},\, \boldsymbol{x} \in \pi_i) \approx \Phi\left\{\frac{E(\hat{f}(\boldsymbol{x}|\pi_i)) - E(\hat{f}(\boldsymbol{x}|\pi_2))}{\sqrt{V(\hat{f}(\boldsymbol{x}|\pi_1)) + V(\hat{f}(\boldsymbol{x}|\pi_2))}}(-1)^i\right\}, \qquad i = 1, 2 \tag{3}$$

where $\Phi\{u\} = (1/\sqrt{2\pi})\int_{-\infty}^{u} e^{-(t^2/2)}\,dt$ is the standard function of the Gaussian distribution $N(0, 1)$.

Let us consider the model of Gaussian data with a common covariance matrix (MGCC) with parameters μ_i and Σ. The conditional mean and the variance of the PW density estimate [8] (conditioned at a fixed point $\boldsymbol{x}$) *with respect to all possible training sets*, which consist of M observations, are

$$\begin{aligned} E(\hat{f}_{\mathrm{PW}}(\boldsymbol{x}|\pi_i)) &= \frac{1}{M}\sum_{j=1}^{M}\int N\left(\boldsymbol{X}_j^{(i)}, \mu_i, \Sigma\right) N\left(\boldsymbol{x}, \boldsymbol{X}_j^{(i)}, \lambda^2\boldsymbol{I}\right) d\boldsymbol{X}_j^{(i)} \\ &= N\left(\boldsymbol{x}, \mu_i, \Sigma + \lambda^2\boldsymbol{I}\right) \end{aligned}$$

and

$$\begin{aligned} V(\hat{f}_{\mathrm{PW}}(\boldsymbol{x}|\pi_i)) &= \frac{1}{M}\left[\frac{\left|2\Sigma + \lambda^2\boldsymbol{I}\right|^{1/2}}{\lambda^p}\left(N\left(\boldsymbol{x}, \mu_i, 2\Sigma + \lambda^2\boldsymbol{I}\right)\right)^2 \right. \\ &\qquad \left. - \left(E(\hat{f}_{\mathrm{PW}}(\boldsymbol{x}|\pi_i))\right)^2\right]. \end{aligned}$$

Let $\boldsymbol{T}$ be a $p \times p$ orthonormal matrix such that $\boldsymbol{T}\Sigma\boldsymbol{T}' = \boldsymbol{D}$ ($\boldsymbol{D}$ is a diagonal matrix of eigenvalues with elements $d_1, d_2, \cdots, d_p$). Then

$$\begin{aligned} V\hat{f}_{\mathrm{PW}}(\boldsymbol{x}|\pi_i) &= \frac{1}{M}\left[\prod_{j=1}^{p}\sqrt{1 + \frac{2d_j}{\lambda^2}}\left(N\left(\boldsymbol{x}, \mu_i, 2\Sigma + \lambda^2\boldsymbol{I}\right)\right)^2 \right. \\ &\qquad \left. - \left(E\hat{f}_{\mathrm{PW}}(\boldsymbol{x}|\pi_i)\right)^2\right]. \end{aligned} \tag{4}$$

For $\lambda^2 \to 0$ the variance of the PW density estimate is determined primarily by the term $(1/M)\prod_{j=1}^{p}\sqrt{1 + (2d_j/\lambda^2)}$. This term decreases when the value of the smoothing parameter λ^2 or/and the training sample size M increases. Let the eigenvalues of the covariance matrix Σ be equal: $d_1 = d_2 = \cdots = d_p = d$ and assume that the number of features p is increased. Then in order to keep the variance (4) constant, the training sample size M should increase exponentially with p

$$M \equiv \left(1 + \frac{2d}{\lambda^2}\right)^{p/2}. \tag{5}$$

Let us assume now that several eigenvalues of the covariance matrix Σ are very small $d_1 = d_2 = \cdots = d_r = d$, $d_{r+1} = d_{r+2} = \cdots = d_p = \varepsilon \to 0$. We call the number r *the intrinsic dimensionality* of the data for the MGCC model. For this data model we have instead of (5)

$$M \equiv \left(1 + \frac{2d}{\lambda^2}\right)^{r/2}. \tag{6}$$

It means that *small training set properties of the PW density estimate* (2) *are not determined by the formal data dimensionality p, but by the true—the intrinsic dimensionality r*. Thus the number of training vectors M required to design this classifier should increase exponentially with r.

With an increase in λ the bias of the PW density estimate increases, however the variance decreases. Therefore in order to minimize the classification error one needs to find an optimal value of the smoothing parameter λ.

The PW estimate is a generalization of Rozenblatt's generalized histogram approach [16]. In this approach one calculates a number $m_i(\boldsymbol{x})$ of training vectors from the ith class, which can be found in a nearest neighborhood $\Omega(\boldsymbol{x})$ of the vector $\boldsymbol{x}$. Let us define the nearest neighborhood $\Omega(\boldsymbol{x})$ by a hypercube with width h and volume h^p, and let us analyze the generalized histogram classifier trained by $2MR$ "noisy" training vectors $\boldsymbol{U}_{jr}^{(i)}$. Then the density estimate is $\hat{f}_{\mathrm{R}}(\boldsymbol{x}|\pi_i) = (m_i(\boldsymbol{x})/MRh^p)$, and the classification will be performed according to the numbers $m_1(\boldsymbol{x})$ and $m_2(\boldsymbol{x})$ of the training vectors from classes π_1 and π_2, falling into the cell $\Omega(\boldsymbol{x})$. The number $m_i(\boldsymbol{x}) = \sum_{j=1}^{M} m_{ij}(\boldsymbol{x})$, where $m_{ij}(\boldsymbol{x})$ is the number of "noisy" vectors $\boldsymbol{U}_{jr}^{(i)}$ $(r = \overline{1, R})$ generated around the training vector $\boldsymbol{X}_j^{(i)}$, falling into the cell $\Omega(\boldsymbol{x})$. The probability of a "noisy" vector to fall into the cell $\Omega(\boldsymbol{x})$ with volume h^p is $P_{ij}(h, p) = N(\boldsymbol{x}, \boldsymbol{X}_j^{(i)}, \lambda^2\boldsymbol{I})h^p$. Numbers $m_{ij}(\boldsymbol{x})$ are random ones. For very small values of h the numbers $m_{ij}(\boldsymbol{x})$ are independent binomial random variables with mean $P_{ij}(h, p)R$ and with variance $P_{ij}(h, p)(1 - P_{ij}(h, p))R \approx P_{ij}(h, p)R$. Thus we get

$$\begin{aligned} E\left(\hat{f}_{\mathrm{R}}\left(\boldsymbol{x}\middle|\boldsymbol{X}_1^{(i)}, \boldsymbol{X}_2^{(i)}, \cdots, \boldsymbol{X}_M^{(i)}, \pi_i\right)\right) &= \frac{1}{M}\sum_{j=1}^{M} N\left(\boldsymbol{x}, \boldsymbol{X}_j^{(i)}, \lambda^2\boldsymbol{I}\right) \\ &= \hat{f}_{\mathrm{PW}}(\boldsymbol{x}|\pi_i) \end{aligned}$$

and

$$\begin{aligned} V\left(\hat{f}_{\mathrm{R}}\left(\boldsymbol{x}\middle|\boldsymbol{X}_1^{(i)}, \boldsymbol{X}_2^{(i)}, \cdots, \boldsymbol{X}_M^{(i)}, \pi_i\right)\right) &\approx \frac{1}{M^2Rh^p}\sum_{j=1}^{M} N\left(\boldsymbol{x}, \boldsymbol{X}_j^{(i)}, \lambda^2\boldsymbol{I}\right) \\ &= \frac{1}{MRh^p}\,\hat{f}_{\mathrm{PW}}(\boldsymbol{x}|\pi_i). \end{aligned}$$

We see that for $R \to \infty$, $h \to 0$, and $M^2Rh^p \to \infty$, the histogram approach with NI results in a density estimate, which is equivalent to the PW estimate. Thus, for the PW estimate we have

$$\frac{E(\hat{f}_{\mathrm{PW}}(\boldsymbol{x}|\pi_i))}{\sqrt{V(\hat{f}_{\mathrm{PW}}(\boldsymbol{x}|\pi_i))}} \approx \left(Rh^p\sum_{j=1}^{M} N\left(\boldsymbol{x}, \boldsymbol{X}_j^{(i)}, \lambda^2\boldsymbol{I}\right)\right)^{1/2}. \tag{7}$$

When R is finite, there is an additional random factor which increases the variance of the class conditional density estimate. This can deteriorate the small sample properties of the classifier.

C. Noise Injection

Now let us consider the effect of the intrinsic dimensionality on the effectiveness of NI. Suppose the intrinsic dimensionality is small in the nearest neighborhood $\Omega(\boldsymbol{x})$, i.e., the nearest training vector $\boldsymbol{X}_j^{(i)}$ of the vector $\boldsymbol{x}$ is in a subspace $S(\boldsymbol{x})$ with dimensionality r. The neighborhood $\Omega(\boldsymbol{x})$ of the vector $\boldsymbol{x}$ is defined as a region, where $N(\boldsymbol{x}, \boldsymbol{X}_j^{(i)}, \lambda^2\boldsymbol{I}) > \varepsilon$ with sufficiently small value of ε. The intrinsic dimensionality r is defined by assuming that the $s = p - r$ last components of vectors

$$\begin{aligned}\boldsymbol{Y}_j^{(i)} &= \boldsymbol{T}(\boldsymbol{x})\left(\boldsymbol{x} - \boldsymbol{X}_j^{(i)}\right)\\ &= \left(Y_{j1}^{(i)}, Y_{j2}^{(i)}, \cdots, Y_{jr}^{(i)}, Y_{jr+1}^{(i)}, \cdots, Y_{jp}^{(i)}\right)'\end{aligned}$$

are equal or very close to zero. $\boldsymbol{T}(\boldsymbol{x})$ is an orthonormal transformation matrix.

Under the subspace assumption the vector $\boldsymbol{x}$ to be classified is located in the subspace $S(\boldsymbol{x})$, where the last s components of the vector $\boldsymbol{y} = \boldsymbol{T}(\boldsymbol{x})\boldsymbol{x}$ are equal or very close to zero. Let us denote

$$\begin{aligned}\boldsymbol{Y}_{j1}^{(i)} &= \left(Y_{j1}^{(i)}, Y_{j2}^{(i)}, \cdots, Y_{jr}^{(i)}\right)'\\ \boldsymbol{Y}_{j2}^{(i)} &= \left(Y_{jr+1}^{(i)}, \cdots, Y_{jp}^{(i)}\right)'\\ \boldsymbol{y}_1 &= (y_1, y_2, \cdots, y_r)'\\ \boldsymbol{y}_2 &= (y_{r+1}, \cdots, y_p)'.\end{aligned}$$

Then

$$\begin{aligned}P_{ij}(h, p) &= N\left(\boldsymbol{x}, \boldsymbol{X}_j^{(i)}, \lambda^2\boldsymbol{I}\right)h^p\\ &= N\left(\boldsymbol{y}_1, \boldsymbol{Y}_{j1}^{(i)}, \lambda^2\boldsymbol{I}\right)N\left(\boldsymbol{y}_2, \boldsymbol{Y}_{j2}^{(i)}, \lambda^2\boldsymbol{I}\right)h^p.\end{aligned}$$

Under the subspace assumption $\boldsymbol{y}_2 - \boldsymbol{Y}_{j2}^{(i)} \approx 0$. Thus,

$$P_{ij}(h, p) = \left(\frac{h}{\sqrt{2\pi}\lambda}\right)^s h^r N\left(\boldsymbol{y}_1, \boldsymbol{Y}_{j1}^{(i)}, \lambda^2\boldsymbol{I}\right)$$

and

$$\begin{aligned}&E\left(\hat{f}_{\mathrm{R}}\left(\boldsymbol{x}\middle|\boldsymbol{X}_1^{(i)}, \boldsymbol{X}_2^{(i)}, \cdots, \boldsymbol{X}_M^{(i)}, \pi_i\right)\right)\\ &\quad= \frac{1}{M}\left(\frac{1}{\sqrt{2\pi}\lambda}\right)^s \sum_{j=1}^{M} N\left(\boldsymbol{y}_1, \boldsymbol{Y}_{j1}^{(i)}, \lambda^2\boldsymbol{I}\right)\end{aligned}$$

and

$$\begin{aligned}&V\left(\hat{f}_{\mathrm{R}}\left(\boldsymbol{x}\middle|\boldsymbol{X}_1^{(i)}, \boldsymbol{X}_2^{(i)}, \cdots, \boldsymbol{X}_M^{(i)}, \pi_i\right)\right)\\ &\quad\approx \frac{1}{M^2Rh^p}\left(\frac{1}{\sqrt{2\pi}\lambda}\right)^s \sum_{j=1}^{M} N\left(\boldsymbol{y}_1, \boldsymbol{Y}_{j1}^{(i)}, \lambda^2\boldsymbol{I}\right).\end{aligned}$$

Following (3), in the analysis of the conditional generalization error $P(\text{misclassification}|\boldsymbol{x}, \boldsymbol{x} \in \pi_i)$ we are interested in the ratio

$$\frac{E(\hat{f}_{\mathrm{R}}(\boldsymbol{x}|\pi_i))}{\sqrt{V(\hat{f}_{\mathrm{R}}(\boldsymbol{x}|\pi_i))}}.$$

Under the subspace assumption we obtain

$$\frac{E(\hat{f}_{\mathrm{R}}(\boldsymbol{x}|\pi_i))}{\sqrt{V\hat{f}_{\mathrm{R}}(\boldsymbol{x}|\pi_i)}} \approx \left(R^*h^r \sum_{j=1}^{M} N\left(\boldsymbol{y}_1, \boldsymbol{Y}_{j1}^{(i)}, \lambda^2\boldsymbol{I}\right)\right)^{1/2} \tag{8}$$

where $R^* = (h/\sqrt{2\pi}\lambda)^s R$. The parameter R^* is called the *effective* number of noise injections. Comparing (8) with (7) indicates that for $h < \lambda$ the factor $(h/\sqrt{2\pi}\lambda)^s$ reduces the influence of the number of noise injections R. Consequently, the generalization error of the histogram classifier with NI increases in comparison with the PW classifier. In order to reduce the generalization error *we need to increase* R. An alternative way is to use instead of spherical noise the "directed" ("singular") noise with a small or zero value of λ in directions of zero eigenvalues of the conditional covariance matrix $\boldsymbol{S}_M(\boldsymbol{x})$ in the nearest neighborhood of the vector $\boldsymbol{x}$. Small values of λ in $s = p - r$ directions increase $(h/\sqrt{2\pi}\lambda)^s$ and do not destroy the intrinsic dimensionality of the data. One of the possibilities is to estimate the covariance matrix $\boldsymbol{S}_M(\boldsymbol{x})$ and to use its eigenvectors to determine the directions of NI. This approach has been used in a prediction algorithm ZET [17]. An alternative is to use k-NN DNI as suggested by Duin [5].

The approach of k-NN DNI consists in generating noise only in the direction of the k-nearest neighbors from the same pattern class of the object under consideration. Let $\boldsymbol{X}_j^{(i)}$ be an object under consideration and $\boldsymbol{Q}_{j1}^{(i)}, \boldsymbol{Q}_{j2}^{(i)}, \cdots, \boldsymbol{Q}_{jK}^{(i)}$ be its K-nearest neighbors from the same pattern class. We determine the k-NN directed noise vectors as

$$\boldsymbol{Z}_{jr}^{(i)} = \lambda \times \frac{1}{K}\sum_{k=1}^{K} \xi_k\left(\boldsymbol{X}_j^{(i)} - \boldsymbol{Q}_{jk}^{(i)}\right)$$

where $\xi_k \sim N(0, 1)$, K is the number of the nearest neighbors used, and λ is a scaling parameter.

The idea of using k-NN directed noise is based on the assumption that in generating noise around a training object, it is useful to take into consideration the distribution of the k-nearest neighbors of this object, especially in the case of a low intrinsic data dimensionality, when data objects lie in the subspace of the feature space. The k-NN DNI makes the training data set more complete in a somewhat different way than bootstrapping [19] does (no data copies are made). It does not destroy the low intrinsic data dimensionality, as the new objects are generated in the subspace determined by the k-nearest neighbors. Therefore, in comparison with spherical Gaussian NI it is less likely that the k-NN DNI deteriorates the training set. In addition, the k-NN DNI is more "economical": one adds noise only in useful directions, saving computer time and memory. Another advantage of the k-NN directed noise in comparison with ZET and similar algorithms, based on the normal data distribution assumption,

is that it does not depend on knowledge of the data distribution. Therefore, the k-NN DNI could be successfully applied to the data of any distribution (e.g., not normally distributed or clustered data) if it is known that intrinsic data dimensionality is much smaller than the feature size.

Thus the conclusion follows: *The number of noise injections R as well as the directions of noise injection are very important factors which determine an effectiveness of the noise injection technique in the training of the statistical classifiers discussed above*.

III. Noise Injection in Multilayer Perceptron Training

Noise injection, when noise vectors are normally distributed $N(\mathbf{0}, \lambda^2\boldsymbol{I})$, is equivalent to the ridge-estimate of the covariance matrix in the standard FLD [4], as well to weight decay in linear SLP training [3], [26], [27] and is similar to the smoothing in the PW classifier. NI helps to fill in gaps between training objects, smoothing the generalization error and stabilizing the training procedure [4], [6], [7], [28]. In the previous section it was shown that the intrinsic data dimensionality influences the effectiveness of NI for statistical classifiers. When the intrinsic data dimensionality r is small, more noise injections R are required in order to reduce the generalization error. One of the ways to reduce R is to add noise only in useful directions (the k-NN DNI), ignoring the directions where no data are located. It is reasonable to suppose that similar effects are also valid for MLP's. As theoretical analysis for MLP's requires multiple approximations and leads to tedious algebra, a simulation is used in order to confirm for MLP's the conclusions made for statistical classifiers.

In all our simulations we have used the perceptron with three hidden units which was trained by the Levenberg–Marquardt learning rule [9]. All results were averaged over ten independent training sets and five different initializations (50 independent experiments in total) with an exception for a three-dimensional sinusoidal dataset. For these data ten independent training sets and ten different initializations were used (100 independent experiments in total). NI on the training set was performed in such a way that in each training sweep the training data set was exchanged by R noise vectors generated from the original training set vectors. In our experiments we chose $R = 100$, for each training set size. The variance for Gaussian NI and the scaling parameter λ for k-NN DNI were optimized separately on each training data set with respect to the smallest apparent error.

A. Data

Four artificial data sets, concentrated in a subspace of the feature space, are used in our experimental investigations.

The first data set consists of two eight-dimensional classes. The first two features of the data classes are uniformly distributed with unit variance spherical Gaussian noise along two $2\pi/3$ concentric arcs with radii 6.2 and 10.0 for the first and the second class, respectively

$$\vec{X}^{(1)} = \begin{pmatrix} X_1^{(1)} \\ X_2^{(1)} \end{pmatrix} = \begin{pmatrix} \rho_1 \cos(\gamma_1) + \xi_1 \\ \rho_1 \cos(\gamma_1) + \xi_2 \end{pmatrix}$$

$$\vec{X}^{(2)} = \begin{pmatrix} X_1^{(2)} \\ X_2^{(2)} \end{pmatrix} = \begin{pmatrix} \rho_2 \cos(\gamma_2) + \xi_3 \\ \rho_2 \cos(\gamma_2) + \xi_4 \end{pmatrix}$$

where

$$\rho_1 = 6.2, \quad \rho_2 = 10, \quad \xi_i \sim N(0, 1), \quad \gamma_k \sim U\left(-\frac{\pi}{3}, \frac{\pi}{3}\right), <, \quad i = \overline{1, 4};\ k = 1, 2.$$

The other six features have the same spherical Gaussian distribution with zero mean and variance 0.1 for both classes. We will call these data "*banana-shaped data*" (BSD).

The second data set consists also of two eight-dimensional banana-shaped classes. The first two features of the data classes are the same as in the data described above. But the last six features have been generated as $x_j = x_2^2 + 0.001 \cdot \xi_j, \xi_j \sim N(0, 1)$, $j = \overline{3, 8}$ in order to make the local intrinsic dimensionality of this data set equal to two.

The third data set consists of two three-dimensional classes. In the first two features both data classes have the same Gaussian distribution $N\left(\mathbf{0}, \left[\begin{smallmatrix} 40 & -5 \\ 30 & 1 \end{smallmatrix}\right]\right)$. The third feature has been generated as a function of the first feature $x_3 = \sin(x_1/\pi)$ and $x_3 = \sin(x_1/\pi) + 0.1$ for the first and second data class, respectively. This data set will be called "*sinusoidal data*" (SD).

The fourth data set consists of two 12-dimensional classes. Vectors in each pattern class π_i, $i = 1, 2$, are uniformly distributed in the first two features space across the line $x_1 = (1/\sqrt{2})x_2 + 0.01 \cdot (-1)^i$. Another ten data features are generated as $x_j = x_2^2 + 0.001 \cdot \xi_j, \xi_j \sim N(0, 1), j = \overline{3, 12}$. By this the local intrinsic dimensionality of this data set is equal to two. We will call these data "*uniformly distributed data*" (UDD).

B. The Effect of the Intrinsic Data Dimensionality on Noise Injection

Let us consider the performance of the perceptron without NI, with Gaussian NI and with k-NN DNI for all four data sets described above. Simulation results are presented in Table I.

By considering two sets of eight-dimensional BSD, one with high and another one with low intrinsic dimensionality it can be seen that the intrinsic data dimensionality influences, both, the classifier performance and the effectiveness of NI. When the data training set is small and situated in a subspace of the feature space, the perceptron tries to build the discriminant function in the feature space without taking into account the intrinsic data dimensionality. By this reason the number of local minima of the pattern cost function arises together with the chance to be trapped into the bad local minima (with a high generalization error). Therefore, the low intrinsic dimensionality of the data can cause a worse performance of the perceptron. As NI smooths the surface of the perceptron cost function, one gets less deep local minima and a more stable classifier. The small sample size properties of such a classifier depend on the intrinsic data dimensionality. The perceptron with NI has a better performance for the data with the smaller intrinsic dimensionality. In other words, *when the data intrinsic dimensionality is smaller, a*

TABLE I
THE MEAN GENERALIZATION ERRORS OF THE PERCEPTRON WITHOUT NI, WITH GAUSSIAN NI AND k-NN DIRECTED NOISE INJECTION (k = 2, 3, 4, 5) AND THEIR STANDARD DEVIATIONS VERSUS THE NUMBER OF THE TRAINING SAMPLES PER CLASS

for 8-dimensional banana-shaped data with high intrinsic dimensionality						
method	M=6	M=10	M=20	M=50	M=100	M=150
without NI	0.42 (0.009)	0.36 (0.015)	0.26 (0.015)	0.163 (0.012)	0.107 (0.005)	0.078 (0.002)
GNI	0.23 (0.012)	0.18 (0.008)	0.18 (0.008)	0.133 (0.006)	0.107 (0.005)	0.096 (0.003)
2-NN DNI	0.37 (0.011)	0.27 (0.009)	0.21 (0.007)	0.136 (0.004)	0.097 (0.003)	0.09 (0.003)
3-NN DNI	0.35 (0.010)	0.28 (0.009)	0.20 (0.007)	0.13 (0.005)	0.094 (0.002)	0.086 (0.004)
4-NN DNI	0.35 (0.009)	0.28 (0.007)	0.19 (0.006)	0.14 (0.004)	0.094 (0.003)	0.087 (0.003)
5-NN DNI	0.37 (0.009)	0.28 (0.008)	0.20 (0.007)	0.13 (0.004)	0.092 (0.003)	0.088 (0.005)
for 8-dimensional banana-shaped data with low intrinsic dimensionality						
method	M=6	M=10	M=20	M=50	M=100	M=150
without NI	0.45 (0.011)	0.43 (0.015)	0.30 (0.027)	0.17 (0.026)	0.09 (0.014)	0.064 (0.009)
GNI	0.23 (0.021)	0.12 (0.015)	0.064 (0.010)	0.051 (0.002)	0.05 (0.001)	0.047 (0.001)
2-NN DNI	0.14 (0.017)	0.093 (0.005)	0.075 (0.009)	0.054 (0.002)	0.052 (0.001)	0.049 (0.001)
3-NN DNI	0.12 (0.013)	0.077 (0.006)	0.07 (0.003)	0.057 (0.001)	0.052 (0.002)	0.049 (0.001)
4-NN DNI	0.11 (0.010)	0.077 (0.005)	0.078 (0.008)	0.054 (0.001)	0.052 (0.001)	0.050 (0.001)
5-NN DNI	0.10 (0.013)	0.095 (0.011)	0.068 (0.004)	0.056 (0.001)	0.052 (0.002)	0.049 (0.001)
for 3-dimensional sinusoidal data with low intrinsic dimensionality						
method	M=10	M=20	M=50	M=100	M=300	M=1000
without NI	0.39 (0.008)	0.30 (0.011)	0.18 (0.016)	0.14 (0.017)	0.14 (0.014)	0.10 (0.011)
GNI	0.39 (0.008)	0.30 (0.013)	0.20 (0.015)	0.18 (0.016)	0.15 (0.013)	0.16 (0.016)
2-NN DNI	0.35 (0.010)	0.21 (0.013)	0.14 (0.015)	0.096 (0.011)	0.11 (0.013)	0.12 (0.012)
3-NN DNI	0.36 (0.014)	0.23 (0.013)	0.14 (0.013)	0.10 (0.013)	0.12 (0.013)	0.13 (0.015)
4-NN DNI	0.35 (0.008)	0.23 (0.013)	0.12 (0.012)	0.096 (0.012)	0.11 (0.014)	0.12 (0.013)
5-NN DNI	0.35 (0.009)	0.23 (0.012)	0.14 (0.014)	0.087 (0.011)	0.11 (0.013)	0.10 (0.012)
for 12-dimensional uniform distributed data with low intrinsic dimensionality						
method	M=6	M=10	M=20	M=50	M=100	M=150
without NI	0.40 (0.02)	0.14 (0.02)	0.037 (0.015)	0.003 (0.003)	0.0005 (0.0003)	0.0003 (0.0003)
GNI	0.38 (0.02)	0.10 (0.02)	0.036 (0.011)	0.001 (0.001)	0.0007 (0.0004)	0.0006 (0.0002)
2-NN DNI	0.20 (0.025)	0.08 (0.015)	0.029 (0.006)	0.002 (0.001)	0.0005 (0.0003)	0.0008 (0.0003)
3-NN DNI	0.18 (0.023)	0.08 (0.013)	0.026 (0.006)	0.002 (0.001)	0.0005 (0.0001)	0.0002 (0.0001)
4-NN DNI	0.23 (0.023)	0.077 (0.012)	0.027 (0.006)	0.002 (0.001)	0.0005 (0.0001)	0.0005 (0.0001)
5-NN DNI	0.19 (0.026)	0.095 (0.012)	0.03 (0.006)	0.002 (0.001)	0.0005 (0.0001)	0.0015 (0.0007)

smaller training set size is required to achieve by noise injection the same results as for the data with high intrinsic dimensionality.

Also it can be observed that the perceptron with k-NN DNI outperforms the perceptron with Gaussian NI for the data with low intrinsic dimensionality for the small training sample sizes. This does not happen for the data with high intrinsic dimensionality. We see, that *the directions of NI become important for small sample sizes when the intrinsic dimensionality of the data is small*.

Surprisingly the results obtained for different numbers (k = 2, 3, 4, 5) of nearest neighbors in k-NN DNI are similar. When the training set is small, rather often it misrepresents the distribution of the entire data set. Any training object could be misleading, giving wrong directions for Gaussian NI. Appending an additional nearest neighbor could worsen the situation. On the other hand, large training sets represent the distribution of the entire data set more accurately. Thereby, two nearest neighbors of the object are as informative as three or more nearest neighbors. In any case the directions for Gaussian NI are defined cor-

TABLE II
THE MEAN GENERALIZATION ERRORS OF THE PERCEPTRON WITH GAUSSIAN AND k-NN DIRECTED NOISE INJECTION AND THEIR STANDARD DEVIATIONS VERSUS THE NUMBER OF NOISE INJECTIONS

for 8-dimensional banana-shaped data with low intrinsic dimensionality and with the training sample size per class equal to 50					
method	R=10	R=15	R=25	R=50	R=100
GNI	0.299 (0.028)	0.233 (0.027)	0. 141 (0.02)	0.071 (0.007)	0.05 (0.001)
2-NN DNI	0.306 (0.026)	0.203 (0.025)	0.114 (0.02)	0.054 (0.001)	0.055 (0.001)
for 12-dimensional uniformly distributed data with low intrinsic dimensionality and with the training sample size per class equal to 20					
method	R=10	R=15	R=25	R=50	R=100
GNI	0.20 (0.026)	0.10 (0.023)	0.04 (0.014)	0.005 (0.002)	0.03 (0.011)
2-NN DNI	0.06 (0.015)	0.025 (0.006)	0.01 (0.004)	0.017 (0.006)	0.019 (0.005)

rectly. Therefore, *the effectiveness of the k-NN DNI does not depend on the number k of nearest neighbors*. However, it should be realized that our method contains a built-in optimization of the variance of the noise, which may compensate a possible influence of k.

As the effectiveness of the k-NN DNI does not depend on the number k of nearest neighbors, we have chosen $k = 2$ for k-NN DNI in all other experiments.

C. The Effect of the Number of Noise Injections on the Efficiency of Noise Injection

Let us compare the performance of the perceptron with Gaussian spherical and 2-NN directed NI versus the number of noise injections R for two data sets with low intrinsic dimensionality: eight-dimensional BSD with an intrinsic dimensionality of two and with the training sample size per class equal to 50 and 12-dimensional UDD with an intrinsic dimensionality of two and with the training sample size per class equal to 20.

Simulation results (Table II) show the following.

1) Generalization error decreases with an increase in the number of noise injections R and stops decreasing when $R \to \infty$. It conforms the theoretical conclusions obtained for parametric and nonparametric statistical classifiers discussed in Section II.
2) The perceptron with 2-NN DNI has a better performance. We see, that for small numbers of noise injections R the perceptron with 2-NN DNI outperforms the perceptron with Gaussian spherical NI, because in 2-NN DNI Gaussian noise is generated only in useful directions, in which the data are located. For large values of the number of noise injections R the results for both NI methods become similar. For very large values of R the spherical Gaussian noise vectors $\boldsymbol{Z}_{jr}^{(i)} \sim N(\boldsymbol{0}, \lambda^2 \boldsymbol{I})$ $(r = 1, 2, \cdots, R)$ become symmetrically distributed in all directions around a training vector $\boldsymbol{X}_j^{(i)}$. Therefore the negative effect of a finite value of R vanishes.

The considered examples demonstrate, that *the effectiveness of NI in perceptron training depends not only on the type of noise injection, but also on the number of noise injections R.*

IV. CONCLUSIONS

Theoretical results obtained for one parametric and two nonparametric statistical classifiers and simulation study carried out for MLP's show the following.

1) Noise injection acts as a regularization factor which helps to reduce the generalization error.
2) There exists an optimal value of the noise variance λ.
3) The number of noise injections R should be sufficiently large. However, too large values of R do not diminish the generalization error and increase computing time.
4) The necessary number of noise injections R^* depends on the data dimensionality p.
5) If the intrinsic data dimensionality r is small, R^* can be reduced by avoiding adding noise to "unnecessary" directions. The k-NN directed NI is one of possible effective means to do this.

REFERENCES

[1] A. K. Jain and B. Chandrasekaran, "Dimensionality and sample size considerations in pattern recognition practice," in *Handbook of Statistics*, P. R. Krishnaiah and L. N. Kanal, Eds. Amsterdam: North-Holland, 1987, vol. 2, pp. 835–855.
[2] Š. Raudys and A. K. Jain, "Small sample size effects in statistical pattern recognition: Recommendations for practitioners," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, pp. 252–264, 1991.
[3] K. Matsuoka, "Noise injection into inputs in back-propagation learning," *IEEE Trans. Syst., Man, Cybern.*, vol. 22, pp. 436–440, 1992.
[4] Š. Raudys, M. Skurichina, T. Cibas, and P. Gallinari, "Optimal regularization of neural networks and ridge estimates of the covariance matrix in statistical classification," *Pattern Recognition Image Anal.: Advances Math. Theory Applicat. (Int. J. Russian Academy Sci.)*, vol. 5, no. 4, pp. 633–650, 1995.
[5] R. P. W. Duin, "Nearest neighbor interpolation for error estimation and classifier optimization," in *Proc. 8th Scandinavian Conf. Image Anal.*. Tromso, Norway, 1993, pp. 5–6.
[6] M. Skurichina and R. P. W. Duin, "Bagging for linear classifiers," *Pattern Recognition*, vol. 31, no. 7, pp. 909–930, 1998.
[7] H. Sakagushi, "Stochastic dynamics and learning rules in layered neural networks," *Progress Theoretical Phys.*, vol. 83, no. 4, pp. 693–700, Apr. 1990.
[8] K. Fukunaga, *Introduction to Statistical Pattern Recognition*. San Diego, CA: Academic, 1990.
[9] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Clarendon, 1995.
[10] R. P. W. Duin, "On the accuracy of statistical pattern recognizers," Ph.D. dissertation, Delft Univ. Technol., 1978.

[11] Š. Raudys, "On determining the training sample size of linear classifier," *Comput. Syst.*, vol. 28, pp. 79–87, 1967.
[12] ——, "On the problems of sample size in pattern recognition," in *Proc. 2nd All-Union Conf. Statist. Methods Contr. Theory*. Moscow, Russia, 1970, pp. 64–67.
[13] ——, "On the effectiveness of Parzen window classifier," in *Informatica*. Vilnius, Lithuania: MII, 1991, vol. 2, pp. 434–454.
[14] ——, "On the shape of pattern error function, initializations and intrinsic dimensionality in ANN classifier design," *Informatica*, vol. 3, no. 3–4, pp. 360–383, 1993.
[15] Š. Raudys and M. Skurichina, "Small sample properties of ridge estimate of the covariance matrix in statistical and neural-net classification," in *New Trends Probability Statist. 3: Multivariate Statist. Matrices Statistics, Proc. 5th Tartu Conf.*, Tartu-Puhajarve, Estonia, May 1994, pp. 237–245.
[16] M. Rozenblatt, "Remarks on some nonparametric estimates of a density function," *Ann. Math. Statist.*, vol. 27, pp. 832–837, 1956.
[17] N. G. Zagoruiko, V. N. Elkina, and V. S. Temirkaev, "ZET—An algorithm of filling gaps in experimental data tables," *Comput. Syst.*, vol. 67, pp. 3–28, 1976.
[18] L. N. Kanal and B. Chandrasekaran, "On dimensionality and sample size in statistical pattern classification," in *Proc. NEC*, vol. 24, 1971, pp. 2–7.
[19] B. Efron and R. Tibshirani, *An Introduction to the Bootstrap*. New York: Chapman and Hall, 1993.
[20] A. E. Hoerl and R. W. Kennard, "Ridge-regression: Biased estimation of nonorthogonal problems," *Technometrics*, vol. 12, pp. 55–67, 1970.
[21] D. Plaut, S. Nowlan, and G. Hinton, "Experiments on learning by back-propagation," Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-CS-86-126, 1986.
[22] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. Eugenics*, pt. II, vol. 7, pp. 179–188, 1936.
[23] E. Parzen, "On estimation of a probability density function and mode," *Ann. Math. Statist.*, vol. 33, pp. 1065–1076, 1962.
[24] J. H. Friedman, "Regularized discriminant analysis," *J. Amer. Statist. Assoc. (JASA)*, vol. 84, pp. 165–175, 1989.
[25] J. Sjoberg and L. Ljung, "Overtraining, regularization and searching for a minimum in neural networks," Dept. Elect. Eng., Linkoping Univ., Sweden, Tech. Rep., 1991.
[26] C. M. Bishop, "Training with noise is equivalent to Tikhonov regularization," *Neural Comput.*, vol. 7, no. 1, pp. 108–116, 1995.
[27] T. K. Leen, "From data distributions to regularization in invariant learning," *Neural Comput.*, vol. 7, no. 5, pp. 974–981, 1995.
[28] Y. Grandvalet, S. Canu, and S. Boucheron, "Noise injection: Theoretical prospects," *Neural Comput.*, vol. 9, no. 5, pp. 1093–1108, 1997.