# Improving Classification via Reconstruction[*]

**Inna Stainvas**
School of Computer Science
Tel-Aviv University
Ramat-Aviv, 69978 ISRAEL.
stainvas@math.tau.ac.il

**Nathan Intrator** [†]
School of Computer Science
Tel-Aviv University
Ramat-Aviv, 69978 ISRAEL.
nin@math.tau.ac.il

**Amiram Moshaiov**
Department of Solid Mechanics,
Materials, and Structures
The Iby and Aladar Fleischman
Faculty of Engineering
Tel-Aviv University
moshaiov@eng.tau.ac.il

July 2000

[*] This work was partially supported by the Israel Science Office, the Israel Science Foundation founded by The Israel Academy of Sciences and Humasnities – Center of Excellence Program, and by the Hermann Minkowski – Minerva Center for Geometry at Tel Aviv University. I. S. was supported by the Don and Sara Marejn Scholarship Fund for Students. Part of this work was done while N. I. was affiliated with the Institute for Brain and Neural Systems at Brown University and supported in part by ONR grants N00014-98-1-0663 and N00014-99-1-0009.

[†] Corresponding author. Address: Box 1843, Brown University, Providence, RI 02912, Phone 401-863-3857, Fax 401-863-3494

### Abstract

Learning a many-parameter model is generally an under-constrained problem that requires additional regularization. We propose to use reconstruction as a regularization constraint for image classification. We show that fusing the two models together is an effective regularizer which adds to the improvement achieved by weight decay constraints. This regularization is effective for single networks and network ensembles.

Classification results are demonstrated on two facial data-sets which are extended to include various image degradations. We show that combining weight decay and reconstruction constraints improves image classification for a wide range of degradations. In particular, "bagging" ensembles which are composed of regularized networks trained on different cross-folds produce best results.

*Keywords:* Reconstruction/Classification network; Dimensionality reduction; Face classification; Network ensembles; Classification of corrupted images; Weight decay; Regularization; Image degradation.

## Contents

## 1 Introduction

With the large increase in computational power, classification from pixel valued images emerges as a topic of considerable research interest. An image is regarded as a vector of pixel intensities and thus, belongs to a very high dimensional space. This leads to parameter estimation difficulties that are widely known as the *curse of dimensionality* (Bellman, 1961), namely, the fact that there is insufficient data to robustly train a classifier in high dimensional space. Training leads to predictors with high variance and in order to control this variance, innovative bias constraints should be used

(Geman et al., 1992). One way is to construct efficient low dimensional representations which are sufficient for the classification task. For example, this can be achieved by multiple class constraints (Intrator and Edelman, 1996).

In this paper, we continue this line of thought and study the effect of regularization in the form of reconstruction constraint on the resulting classifier. We further study if reconstruction constraints can replace, or should be added to weight decay (WD).

Reconstruction constraints have been used for reliability estimation of network familiarity with novel scenes (Pomerleau, 1993) for the purpose of car navigation or for selective attention using the ability of the residual to represent confidence of network familiarity with the input (Baluja and Pomerleau, 1995). A variant of reconstruction constraints was used in regression probelms when some of the inputs were also used as network outputs (Caruana and de Sa, 1997). They suggested a multi task learning (MTL) approach for training networks (Caruana, 1993), however, their approach did not include a regularization between the different task, which we find essential.

Our motivation for using reconstruction as a constraint to classification is the simple fact that this is an essential taks of (visual) cortex, and thus, it may be possible that reconstruction constraints help creating a better hidden representation for classification, when training data size is small. A network performing reconstruction and classification (with no regularization) was proposed to model a portion of hippocampus (Gluck and Myers, 1993).

Our work extends previous work by trying to make recosntruction a practical tool for real-word applications; In particular, we show that regularization plays an important role in finding the optmial constraint level, and that simple weight decay is actually superior to reconstruction constraints when its level is carefuly chosen. We find reconstruction to be effective only in combination with weight decay. We introduce reconstruction constraints into ensemble training and study in detail reconstruction ensembles based on different ensemble parameters. We demonstrate that in this context, bagging ensembles are superior. Finally, we introduce a Bayesian framework for reconstruction ensembles and relate it to MDL modeling.

# 2 Methodology

## 2.1 Statistical motivation

In many parameter models, the bias/variance dilemma (Geman et al., 1992), is more pronounced. When the predictor's bias is appropriate (fits the underlying assumptions about the data), its contribution to the over-all prediction error is often small compared to the contribution of the variance. Different regularization methods exist for finding an optimal tradeoff by means of adding a penalty, usually in the form of some measure of smoothness to the predictor (Wahba, 1990; Poggio and Girosi, 1994).

While smoothness can be considered a universal goal for a predictor, there are other related goals which are based on a measure of the quality of the hidden-units' representation. Some of these constraints are related to the information content of the hidden-unit representation. For example, one might search for a hidden representation which has highest entropy, or which has certain deviations from Gaussian distribution. Furthermore, if one is interested in preserving the cluster structure of the data, a variant of the BCM learning rule (Intrator and Cooper, 1992) can be used, while if high kurtosis is of interest, some kurtosis maximization rule can be used, (Oja, 1995). In the context of entropy maximization and independent component analysis (ICA), a recent review

(Yang and Amari, 1997) provides a lot of insight and relation to other methods. Reconstruction constraints, however, have not been used in the context of improving hidden units representation for classification task.

### 2.2 Reconstruction constraints

Figure 1 presents the architecture of the hybrid classification/reconstruction network. This network attempts to improve the low dimensional representation by minimizing concurrently the mean squared error (MSE) of the reconstruction and classification outputs. In other words, the network attempts to improve the quality of the hidden layer representation by imposing feature selection which is useful for both tasks: classification and reconstruction[1].

The hybrid learning rule for the hidden layer units is a composition of the errors back-propagated from the reconstruction and classification layers. The relative influence of each of the output layers is determined by a constant $\lambda$ which represents a tradeoff between reconstruction and classification confidences.

A hybrid classification/reconstruction architecture

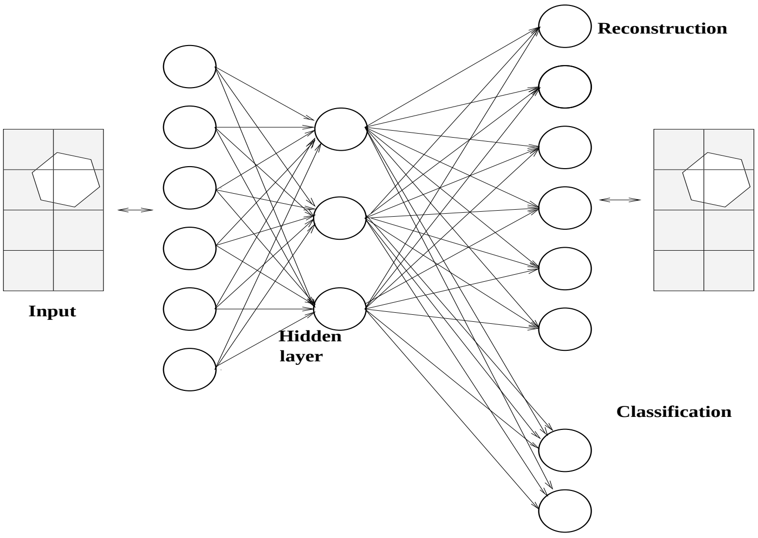

Figure 1: A single hidden layer drives the classification layer and the reconstruction layer. The relative effect of the errors on the gradient that is propagated to the hidden layer is determined by a regularization parameter $\lambda$

### 2.3 Bayesian framework for a hybrid classification/reconstruction networks

When a similar misclassification loss function is applicable to all classes an optimal Bayes classifier assigns class labels to the class with maximal posterior probability (Duda and Hart, 1973). These probabilities are often estimated using parametric models and then plugged-in the Bayesian rule. There are two main paradigms to parameter estimation: *sampling* and *diagnostic* (Ripley, 1996). Both give a parametric model for the joint density $p(x, c|\theta)$ of the feature vector $x$ and class

[1] The hidden layer should have a smaller number of units compared with the inputs, so as to achieve a bottleneck compression and allow for generalization.

label $c$. In the sampling paradigm, interest centers on conditional class densities $p(x|\theta, c)$ and $p(x, c|\theta) = p(x|c, \theta)\pi_c$, with prior class probabilities $\pi_c$ assumed to be known or to be estimated. In the diagnostic paradigm, interest centers on the posterior densities $p(c|x, \theta)$ and:

$$p(x, c|\theta) = p(c|x, \theta)p(x|\theta) \tag{1}$$

In neural network models, the diagnostic paradigm is usually considered and any information about $\theta$ in the unconditional density $p(x|\theta)$ is discarded by conditioning on the observed $x$'s (Bishop, 1995; Ripley, 1996). In our case however, we model both $p(c|x, \theta)$ and $p(x|\theta)$ via the same group of parameters. We show that in the simplest case, this consideration leads to the hybrid classification/reconstruction network introduced above.

In a Bayesian framework, model parameters $\theta$ is found by maximizing the posterior probabilities: $\theta^\star = \arg\max_\theta p(\theta|D)$, where $D$ is a finite training set of pairs $(x_i, c_i)$. Using Bayesian formula: $p(\theta|D) = p(D|\theta)p(\theta)/p(D)$, and since $p(D)$ does not depend on $\theta$, the most plausible model parameters $\theta^\star$ maximize the sum of the log-likelihood of the data $D$ (under the regular assumption of independent samples) and log-priors of the parameters $\theta$:

$$\theta^\star = \arg\max_\theta[\log p(D|\theta) + \log p(\theta)] \tag{2}$$

Assuming the samples to be independent and taking into account (1) we get the following optimization problem:

$$\begin{aligned}\theta^\star = \quad & \arg\max_\theta[\sum_{i=1}^N \log p(x_i, c_i|\theta) + \log p(\theta)] = \\ & \arg\max_\theta[\underbrace{\sum_{i=1}^N \log p(c_i|x_i, \theta)}_{log-likelihood\ \mathcal{L}(c|x,\theta)} + \sum_{i=1}^N \log p(x_i|\theta) + \underbrace{\log p(\theta)}_{\log -prior}]\end{aligned} \tag{3}$$

The first and third RHS terms are recognized as log-likelihood and log-prior, respectively, and are conventionally used to train supervised NN models. In feed-forward networks, typically there is one output unit for each class, and activation of each output unit represents the corresponding posterior probability $p(\mathcal{C}_k|x)$ of the $k$-class. The targets $c$ are often chosen by 1-of-c coding scheme, which assigns $c_{ik} = \delta_{k,cl(i)}$ where $cl(i)$ is the class label of input $x_i$. There are different ways to approximate log-likelihood $\mathcal{L}(c|x, \theta)$ (Bishop, 1995, sections 6.6-6.8):

- sum-of-squares error function $\mathcal{L}(c|x, \theta) = -\beta_1 E_1$ with $E_1 = \frac{1}{K}\sum_{i=1}^N \sum_{k=1}^K (y_k(x_i, \theta) - c_{ik})^2$ and $\beta_1$ induced by analogy with regression tasks, where it is inversely proportional to a noise variance in the outputs.

- cross-entropy error function that models $p(c_i|x_i, \theta) = \prod_{k=1}^K [y_k(x_i, \theta)]^{c_{ik}}$ and $\mathcal{L}(c|x, \theta) = -\sum_{i=1}^N \sum_{k=1}^K c_{ik} \log y_k(x_i, \theta)$

Though the sum-of-squares error is not the most appropriate to classification, it has computational advantage and is widely used (Leung and Zue, 1989; Bishop, 1995).

When some observations are unlabeled they can still be added to the training set $D$. In this case, the summation indices in the first and second terms may be different. When class labels are not present at all, the second RHS term disappears and the plausible model parameters are fit to optimaly reconstruct the input feature vectors $x$. This is essentially unsupervised learning, which, in the simplest case, is realized by a standard autoencoder, or more generally, by minimum description length (MDL) principle (Hinton and Zemel, 1997). For the standard autoencoder the third term may be rewritten as $\log p(x|\theta) = -\beta_2 E_2$, where $E_2$ is a reconstruction error normalized by the number of input units and is given by $E_2 = \frac{1}{d}\sum_{i=1}^{N}\sum_{l=1}^{d}(u_l(x_i,\theta) - x_{li})^2$; $\beta_2$ is a proper normalization coefficient, which is inversely proportional to the noise variance in the reconstruction output. Learning by the optimization (3), corresponds to a flexible combination of supervised and unsupervised learning.

The model parameters $\theta$ are composed of three groups: the hidden weights $w$ shared by reconstruction and classification feed-forward networks and hidden-to-output weights $W_1$ and $W_2$ for classification and reconstruction sublayers. We utilize sigmoidal activation functions in the hidden and classification units and linear functions in the reconstruction units. The optimization task (3) leads to minimization of a goal function:

$$\mathcal{F}(w, W_1, W_2) = \beta_1 E_1(w, W_1) + \beta_2 E_2(w, W_1) - \log p(w, W_1, W_2).$$

Parameters $\beta_1$ and $\beta_2$ are unknown hyperparameters and with a proper normalization can be replaced by a single hyperparameter $\lambda$ with

$$\mathcal{F}(w, W_1, W_2) = (1-\lambda)E_1(w, W_1) + \lambda E_2(w, W_1) - \nu \log p(w, W_1, W_2). \tag{4}$$

Using steepest gradient descent and Gaussian prior for the hidden weights $w$ (with weight decay regularization) (Bishop, 1995) the learning rule can be written as:

$$\begin{aligned} \Delta w &= -\eta((1-\lambda)(\bigtriangledown_w E_1 + \mu w) + \lambda \bigtriangledown_w E_2) \qquad \lambda \in [0,1]. \\ \Delta W_1 &= -\eta(1-\lambda) \bigtriangledown_{W_1} E_1, \quad \Delta W_2 = -\eta\lambda \bigtriangledown_{W_2} E_2. \end{aligned} \tag{5}$$

We rescale the gradients for hidden-to-output weights and consider instead:

$$\Delta W_1 = -\eta \bigtriangledown_{W_1} E_1, \quad \Delta W_2 = -\eta \bigtriangledown_{W_2} E_2. \tag{6}$$

One can easily see that a new rescaled weight increment has positive projection onto the negative gradient of $\mathcal{F}$ and thus amounts to a gradient descent method (Luenberger, 1984).

For $\lambda = 0$ the hidden representation is extracted by a simple classification network with WD smoothing. The weights $W_2$, attempt to reconstruct the input from hidden units that are trained with a classification model only. For $\lambda = 1$ the hidden layer degenerates to an autoencoder hidden layer and the classification task is solved by an independent classification network with the weights $W_1$ from this compressed data representation. For linear activation functions in the hidden layer, the network hidden weights span the space of principal components (Kramer, 1991). This network may be well approximated by a network with sigmoidal activation functions in the hidden layer and WD smoothing. This network is refered to as a PCA network in our results. For $\lambda$ values between 0 and 1, the tradeoff between classification and reconstruction goals affects particularly the hidden representation from which hidden-to-top weights are learned with the same rate independent of $\lambda$.

The notation used for the different cases are given in Table 1.

## 3 Regularization in classification/reconstruction network

There are two regularization parameters in the proposed network. The first $\mu$ is responsible for a smoothness (norm) of the hidden weights and the second $\lambda$ controls the relative influence between weight modification due to classification errors. We apply the following suboptimal (but quicker) sequential optimization:

1. First, an optimal $\mu^\star$ parameter for WD networks is estimated.

2. Second, using $(\mu = \mu^\star)$, an optimal $\lambda^\star$ for Reco.+ WD networks is found.

Our experience shows that the alternating optimization starting from Reco. nets $(\lambda = 0)$ leads to worse results and is, thus, not considered here. We, however, compare regularized Reco. nets with regularized WD and Reco.+WD networks for one of the data-sets used.

### 3.1 Regularization parameter in Bayesian framework

In the Bayesian framework, one has to estimate a posterior distribution of the hyperparameters given the data $p(\zeta|D)$ and prediction of any other variable $\mathcal{P}$ is given by: (Bishop, 1995, Sections 10.4-10.5):

$$p(\mathcal{P}|D) = \int p(\mathcal{P}|D, \zeta)p(\zeta|D)d\zeta, \tag{7}$$

The easiest way to evaluate (7) is known as *evidence approximation* and assumes that the posterior probability distribution $p(\zeta|D)$ for the hyperparameters is sharply picked around $\zeta^\star$ $(p(\zeta|D) \approx \delta(\zeta - \zeta^\star))$ and thus $p(\mathcal{P}|D) \approx p(\mathcal{P}|D, \zeta^\star)$ (MacKay, 1992).

We use the evidence approximation in the form $p(\zeta|D) \approx \frac{1}{S}\sum_1^S \delta(\zeta - \zeta_s^\star)$ which assumes that the posterior probability distribution $p(\zeta|D)$ is sharply picked around several values $\zeta_s^\star$. This assumption combined with an assumption that posterior weights are also well localized leads to averaging over several networks with "good" $\zeta$ values (indeed, we assume that these $\zeta$ are close to each other):

$$p(\mathcal{P}|D) \approx \frac{1}{S}\sum_{s=1}^{S} p(\mathcal{P}|D, \theta_s^\star, \zeta_s^\star) \tag{8}$$

that amounts to neural network ensembles (Section 3.3).

In practice, the regularization parameter $\zeta_s^\star$, is estimated via cross-validation (Section 3.2). We demonstrate that a simple averaging procedure in the vicinity of an optimal $\zeta$ produces better results than seeking a unique optimal $\zeta$ value. We have also attempted to find $p(\zeta|D)$ using "Monte Carlo simulation" as proposed in (Rognvaldsson, 1998)[2]. However, since this procedure does not outperform a simple averaging it is not further discussed.

Network types

| PCA: | Uncons.: | WD: | Reco.: | Reco.+WD: |
|---|---|---|---|---|
| $\lambda = 1$ | $\lambda = 0$ | $\lambda = 0$ | $\lambda > 0$ | $\lambda > 0$ |
| $\mu > 0$ | $\mu = 0$ | $\mu > 0$ | $\mu = 0$ | $\mu > 0$ |

Table 1: We refer to hybrid classification/reconstruction networks with $\lambda = 0$ and without WD $(\mu = 0)$ as *Uncons.* (unconstrained) networks; with $\lambda = 0$ and WD $(\mu \neq 0)$ as *WD* networks; with $\lambda = 1$ and WD as *PCA* networks. Networks with $\lambda$ between marginal values are referred to as *Reco.* networks and when WD is additionally used as *Reco. + WD* networks.

[2] Rognvaldsson estimates an optimal regularization parameter via: $\zeta^\star = (\sum_{k=1}^{K} n_k\zeta_k)/(\sum_{k=1}^{K} n_k)$ where $n_k$ are

### 3.2 Regularization approach via cross-validation

Cross-validation (CV) technique is a well known approach for searching a regularization parameters and is especially useful for small data sets. (Carven and Wahba, 1979; Hastie and Tibshirani, 1990; Wahba, 1990) It is often used for choosing the number of hidden units or deciding about early training stopping in NNs (Bishop, 1995). It proceeds by randomly splitting the data $D$ into $V$-disjoint subsets $D_1, \dots, D_V$ of approximately equal size. For every, $\nu \in V$, a classifier $f^\nu$ is constructed on the data $D \backslash D_\nu$ (the samples not included in $D_\nu$) and a misclassification rate is evaluated on the omitted data subset $D_\nu$: $R(f^\nu) = \frac{1}{|D_\nu|} \sum_{(x_n, c_n) \in D_\nu} \chi(f^\nu(x_n) \neq c_n)$, where $\chi$ is an indicator function, and $|D_\nu|$ is the size of the subset $D_\nu$. It is assumed that the procedure is "stable", i.e. the true classification rate of the classifier $f^\nu$: $R^\star(f^\nu) = P(f^\nu(x) \neq c)$ is nearly equal $R^\star(f)$ where the classifier $f$ is constructed based on the whole $D$. The true rate $R^\star(f)$ is estimated by: $R^{CV}(f) = \frac{1}{V} \sum_{\nu=1}^{V} R(f^\nu)$. A practical recommended number of subsets is around $5 - 10$ (Breiman et al., 1984; Kohavi, 1995). In the context of the regularized neural networks, we consider a set of classifiers that is depends on $\zeta$ and the performance is estimated by $R^{CV}(f_\zeta) = \frac{1}{V} \sum_{\nu=1}^{V} R(f_\zeta^\nu)$.

The optimal parameter ($\zeta^\star$) has to minimize $R^{CV}(f_\zeta)$:

$$\zeta^\star = \arg \min_{\zeta \in \mathcal{Z}} R^{CV}(f_\zeta). \tag{9}$$

For every $\zeta$ and a given cross-fold $D_\nu$, the construction of a classifier $f_\zeta^\nu$ amounts to training a regularized neural networks by gradient descent according to (5-6). Gradient descent in NNs often leads to many local minima. There is no a guarantee that the same optimal $\zeta$ parameter corresponds to all the resulting different models. Therefore, ideally a regularization parameter has to be estimated separately per local minimum or instead, per specific initial weights.

The procedure proceeds as follows; First, all networks are trained with several initial weights for a fixed number of epochs. Then, an optimal parameter $\zeta$ is found based on (9). Finally, the optimal model is tested on an unseen "test set" $T$, a set that has not been used at all during training or model selection.

### 3.3 Regularized Neural Network Ensembles

Another way to assess the effect of regularization constraints is by combining regularized networks into ensembles. It is well known that an ensemble of experts is capable of improving the performance of single experts (Wolpert, 1992; Krogh and Vedelsby, 1995; Raviv and Intrator, 1996). There are two main questions to be addressed when constructing ensembles: (i) how to evaluate an ensemble classification prediction from predictions of its members and (ii) which networks to combine.

There are different ways to evaluate an ensemble classification prediction. The first, is using a majority rule over all the experts in the ensemble (Hansen and Salamon, 1990). We call this a *classification ensemble*. The second rule is based on averaging the real values of the outputs of all the ensemble members and then producing a decision by thresholding. We call this a *regression ensemble*. A method called logarithmic opinion pool (Heskes, 1998), is threshold the direct product of the ensemble members' classification outputs. This is consistent with an interpretation of classification outputs as posterior class probabilities.

found as follows. Average cross-validation errors $e_k$ and variances $\sigma_k$ per each $\zeta_k$ value are found. Assuming errors per $\zeta_k$ to be independent and normal $\mathcal{N}(e_k, \sigma_k^2)$, these $K$ distributions are sampled and a $\zeta$ corresponding to the minimum error is selected as a "winner". This is repeated several times collecting statistics $n_k$ on how often $\zeta_k$ value was a winner. We have used a value $n_k / \sum_{k=1}^{K} n_k$ as an estimate of the posterior probability $p(\zeta_k|D)$.

As our aim is to test the usefulness of regularization constraints and not finding a "optimal" combination of experts, we used only uniform weighing for all ensemble members. Our experiments indicate that a simple averaging with uniform weights improves results and is superior to both classification ensembles and a simple product of experts. Thus, we present results with a simple ensemble averaging.

The improvement in regression ensembles depends on the level of independence of the errors made by the experts. This independence reduces the contribution of the variance portion of the error when ensemble average is used (Raviv and Intrator, 1996). This also gives some hints which networks to combine. We consider three types of simple regression ensembles Ens. A,B and C, with network outputs averaged over: *(A) $\zeta$ values close to the "optimal" $\zeta^\star$*, all networks are trained from the same initial weights (see also Section 3.1); *(B) Initial weights*, all networks correspond to the same degree of regularization (the same regularization parameter $\zeta$); *(C) Different training cross-folds*, all networks correspond to the same regularization values, this is a "bagging" type of ensemble. To enable a comparison of different ensemble types, the number of hybrid networks in all experiments is taken to be the same. The mathematical formulae describing the performance criteria are given in the Appendix.

It turns out that an ensemble of networks that were trained with different regularization parameters $\zeta$ is useful (Ens. A). This result is not obvious since networks that are trained from the same initial weights, data and close $\zeta$ values are likely to be correlated in their outputs. In experiments below, it is found that Ens. B is superior to Ens. A due to its members independence, i.e. due to the existence of many (local) minima in the error surface. Finally, we find that the most powerful are bagging ensembles (Ens. C). Ensembles B and C are also regularized models depending on the regularization parameter. The optimal $\zeta$ for Ens. B is chosen based on cross-validation approach like to single nets and the optimal parameter of Ens. C is taken the same as for Ens. B (see Appendix for details). Ensemble B with $\zeta = 0$ is a conventional ensemble of unconstrained networks that is refered to as *unconstrained* ensemble of type B. We also consider unconstrained bagging ensembles, i.e. ensembles of type C with $\zeta = 0$.

## 4 Image degradation

The accuracy of classification falls abruptly when image quality is slightly degraded. The physical causes of image defects are myriad: natural climate conditions such as fog, rain or snow; partial occlusion and noise; changes in illumination and shadows that are due to movement of surrounding objects. Most of these factors cause image blur, and often image restoration is done to cope with the latter. Restoration does not remove the blur completely and may lead to an additional artificial image enhancement as well.

A real-world classification system has to be robust to realistic image degradation, including degradation process to which it has not been trained for.

Below, we briefly describe the type of degradations that were used and provide examples of degraded images in Figure 2;

**"Undegraded" data:** The original test set (not used for neural network training and regularization parameters tuning) without any image degradation.

**Varying illumination:** Varying illumination is a natural degradation affecting classification performance. It is simulated using a polynomial model (Lai and Fang, 1999): $i'(x, y) = i(x, y) + \beta(x, y)$, $\beta(x, y) = I(sin\phi(x - x_0) - cos\phi(y - y_0))$, where $i(x, y)$ and $i'(x, y)$ are original and arti-

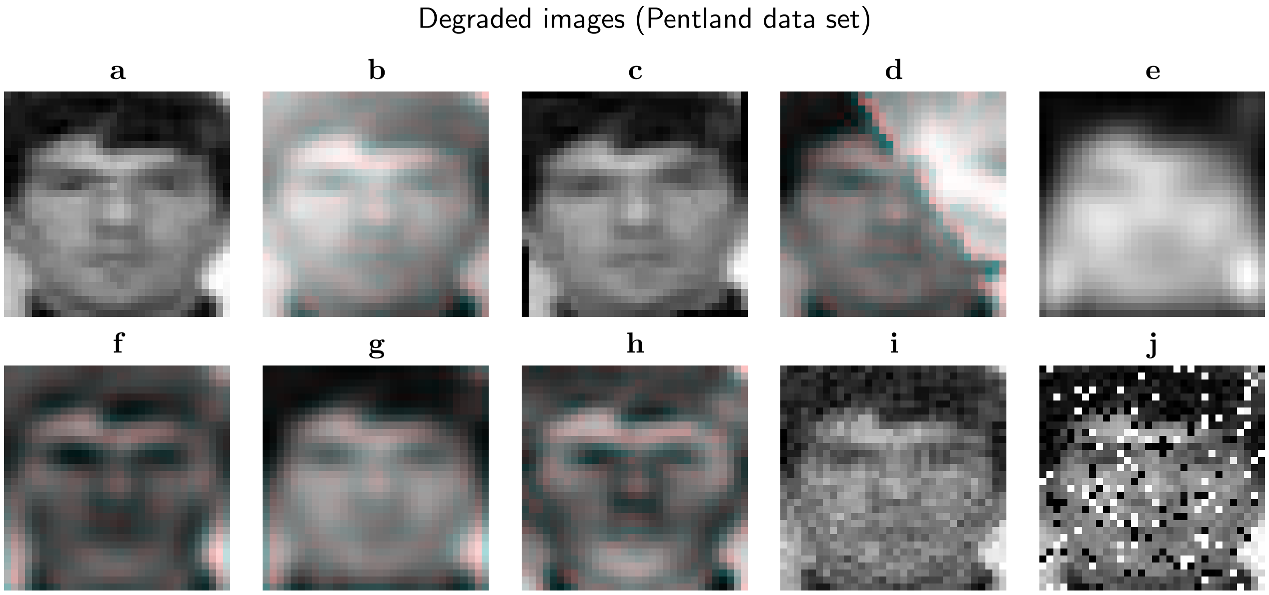

Figure 2: Various image degradations: **a.** undegraded face; **b.** varying illumination; **c.** slightly rotated image; **d.** partially occluded image; **e.** Gaussian blur $\sigma = 2$ pixels; **f.** DOG filter $\sigma_1 = 1$ and $\sigma_2 = 3$ pixels; **g.** Out-of-focus blur $R = 2$; **h.** High-pass-filter $R = 2$; **i.** Gaussian noise $snr = 1$ ; **j.** Salt and Pepper noise $d = 20\%$.

ficially illuminated images, respectively. This illumination does not change the intensity of image pixels lying on a straight line passing through a point $(x_0, y_0)$ in the direction $\phi$. An illumination surface slant variable $I$ is taken to be so that $\max(|\beta(x,y)|) = \max(|i(x,y|)$. Images are illuminated randomly within a small vicinity of the angle $\phi^0$. Our preprocessing (Section 5) partially removes the variability caused by illumination. It appears that the results are more sensitive to the variance of the angle $(\Delta\phi)$ than to the mean value $\phi^0$. This is probably due to our preprocessing, which removes average values.

**Rotation:** The data-set contains faces in orientation and rotation around a frontal horizontal view. The image normalization (Section 5) partially compensates for 3-D rotations at the expense of causing facial distortion.

In order to simulate this type of degradation, images are randomly rotated in the image plane by a small angle chosen uniformly from an interval $[-\alpha,\ \alpha]$.

**Partial occlusion:** This is achieved by replacing the pixel values at any area of arbitrary polygon shape of the face by either the average intensity of the pixels in that area multiplied by some factor $k$ or by the image of the occluding object.

**Blurring with Gaussian filter:** Blurring with Gaussian filter is one of the simplest types of image degradations. We have used a blurring Gaussian with a standard deviation $\sigma = 2$ (pixels). For our data-sets with intermediate resolution this scale of smoothing leads to loss of many details around the eyes and mouth. This is the most difficult transformation for methods that rely on edge detection.

**Blurring with DOG filter:** Difference of Gaussians (DOG) filter, which produces a Mexican hat type receptive field, is a form of image preprocessing known to be present in early mammal vision (Marr, 1982; Kandel and Schwartz, 1991). It enhances edges while smoothing the image. Standard deviations of the on and off center (positive and negative Gaussians) were 1 and 2 (pixels) respectively.

**Out-of-focus blur:** This blur is a common type of degradation when a lens with a circular aperture is defocused. The point spread function (PSF) of this blur is approximated by the cylinder whose radius R depends on the extent of the focus defect (Cannon, 1976).

**High-pass filtering:** This is an ideal high-pass filtering that eliminates small frequencies inside a circle of a small radius R. While this filter is not physically realizable, it is widely used in image processing for performance comparison between different types of filter degradations.

**Gaussian noise:** Gaussian white noise is commonly used to model sensor noise and quantization process (Rosenfeld and Kak, 1982). We limit ourselves to Gaussian noise that acts independently on each pixel with zero mean and some variance that is taken so that a signal to noise ratio is equal to some predefined value $snr$.

**"Salt and Pepper" noise:** This degradation replaces pixel intensities by either the maximum or minimum grey-level value at random locations of a certain percentage of the image (Rosenfeld and Kak, 1982). Results presented here were done with 20% replacement.

# 5 Data-set description and implementation details

The widely available facial data-set (Turk and Pentland, 1993) as well as a face data-set locally collected by the Tel-Aviv University Computer Vision Group (Tankus, 1996) were used in our simulations. While there have been many successful classification approaches to the Turk/Pentland data, we demonstrate that when the images are given in a reduced resolution $32\times32$ (the original and widely-used resolution is $64 \times 64$), or are degraded either by blur or partial occlusion, classification performance deteriorates dramatically. The Turk/Pentland data-set contains 27 images of 15 male faces (we took out the single bearded person). For each person, we randomly choose 15 images for training (data D) and 12 images for testing (data T). The 15 training samples were split into five cross-folds (by taking out 3 different images per person).

Preprocessing details and previous results studying effect of background, illumination and comparison with PCA for original resolution are given in (Intrator et al., 1996). The preprocessing partially removes the variability due to viewpoint, by setting (automatically) the eyes and tip of the mouth to the same position in all images (Tankus et al., 1997). Further preprocessing evaluates the difference between each image and an average over all the training set, leading to the so called "caricature" images (Kirby and Sirovich, 1990).

The second data-set was collected by the Computer Vision group at Tel-Aviv University (TAU). It is of high resolution $84 \times 56$, and contains images of 37 male and female faces with 10 images per person. We compressed images to a low resolution $42 \times 28$ and split data into test T (4 images per person) and training D (6 images per person) sets. Cross-validation with three disjoint groups of size 2 images per person is considered. Preprocessing was similar to the one described above, except that only the eye locations were fixed (Tankus et al., 1997).

All networks used for a given data-set have the same architectural complexity (for the classification part); The number of output units in the reconstruction sublayer is the same as the number of image pixels (1024 for Pentland data and 1176 for TAU data). There are 15 hidden units in all networks and a number of output classification units is as the number of classes (15 for Pentland data-set and 37 for TAU data-set). The initial weights of the networks are chosen randomly out of a uniform distribution between -0.001 and 0.001. A constant learning rate is set to 0.05. Networks for Pentland data- set are trained 3000 epochs and for TAU data set 5000 epochs. Initial weights are resampled 5 times for Pentland data set (the same as a number of training cross-folds in cross-validation) and 4 times for TAU data (the number of training cross-folds is 3).

# 6 Results

To assess the usefulness of reconstruction constraints, we consider single regularized networks and various ensemble combinations (see Section 3.3). Single networks include unconstrained networks, PCA networks and optimal networks regularized either by WD or reconstruction constraints or by both (see Table 1).

Regularized and unconstrained model parameters

| Pentland data-set | | | | |
|---|---|---|---|---|
| **Models** | Nets | Ens. A | Ens. B | Ens. C |
| PCA | $\lambda = 1,\ \mu = 0.05$ | - | $\lambda = 1,\ \mu = 0.05$ | |
| Uncons. | $\lambda = 0,\ \mu = 0$ | - | $\lambda = 0,\ \mu = 0$ | |
| WD | $\lambda = 0$<br>$\mu^\star = 0.05$ | $\lambda = 0$<br>$\mu^\star = 10^{-2} \times [0\ 5\ 10\ 15\ 20]$ | $\lambda = 0$<br>$\mu^\star = 0.05$ | |
| Reco. | $\mu = 0$<br>$\lambda = 0.05$ | $\mu = 0$<br>$\lambda^\star = 10^{-2} \times [0\ 2.5\ 5\ 10\ 15]$ | $\mu = 0$<br>$\lambda = 0.05$ | |
| Reco. +WD | $\mu^\star = 0.05$<br>$\lambda^\star = 0.1$ | $\mu^\star = 0.05$<br>$\lambda^\star = 10^{-2} \times [0\ 1\ 2.5\ 5\ 10]$ | $\mu^\star = 0.05$<br>$\lambda^\star = 0.05$ | |
| TAU data-set | | | | |
| **Models** | Nets | Ens. A | Ens. B | Ens. C |
| PCA | $\lambda = 1,\ \mu = 0.05$ | - | $\lambda = 1,\ \mu = 0.05$ | |
| Uncons. | $\lambda = 0,\ \mu = 0$ | - | $\lambda = 0,\ \mu = 0$ | |
| WD | $\lambda = 0$<br>$\mu^\star = 0.05$ | $\lambda = 0$<br>$\mu^\star = 10^{-3} \times [5\ 10\ 20\ 50]$ | $\lambda = 0$<br>$\mu^\star = 0.005$ | |
| Reco. +WD | $\mu^\star = 0.05$<br>$\lambda^\star = 0.005$ | $\mu^\star = 0.05$<br>$\lambda^\star = 10^{-3} \times [1\ 5\ 10\ 13]$ | $\mu^\star = 0.05$<br>$\lambda^\star = 0.013$ | |

Table 2: Models obtained by considering 4 network architectures (columns) with different degree and type of constraints (rows). There are 5 different networks (column marked by "Nets"): *PCA* has a hidden layer that is similar to PCA representation (see the end of Section 2.3); *Uncons.* stands for unconstrained network; *WD, Reco., Reco. + WD* stand for optimal networks with weight decay, reconstruction and combined reconstruction and WD constraints, respectively. Optimal parameters $\lambda^\star$ and $\mu^\star$ are found using CV approach on data going over all different degradations. *Ens. A–C* stand for ensembles of type A–C, respectively. There are no PCA ensembles of type A and unconstrained nets, since Ens. A is a combination of nets with different regularization parameters, this is marked by (-). For Ens. A with (WD, Reco., Reco. + WD) constraints, the corresponding integration parameters are given. Note, that optimal parameters for networks and ensembles are not the same. All ensembles for Pentland data-set are composed of 5 networks; for TAU data-set ensembles A-B are composed of 4 networks and ensemble C from 3 nets; ensembles generation is explained in Section 3.3.

### 6.1 Model selection considerations

Ideally, model selection has to be specific to the image degradation; instead, it is preferable to have a single model that is moderately good for a wide class of possible degradations (this selection is based on CV approach that is done on degraded data). Optimal model parameters found by CV (see Appendix for details) and integration parameters for ensembles of type A (Ens. A) are given in Table 2. It turns out that the optimal regularization parameters for ensembles and networks are not the same; a similar observation for regression tasks was reported in (Taniguchi and Tresp, 1997). We also note, that an effective area of integration parameters for ensembles of type A, which is taken in a vicinity of the optimal parameters of corresponding networks, turns out to be biased to $\lambda = 0$, i.e. in the direction to unconstrained networks.

Classification results for all degradation types are given in Tables 3 for Pentland data and in Table 4 for TAU data. These tables present averaged error rates and standard deviations obtained from multiple runs with different training sub-sets and initial conditions (see Appendix for details)

### 6.2 Network architecture comparison:

Our results demonstrate that classification is improved most when ensemble averaging is taken over different cross-folds[3] Ens. C.

It follows that averaging over different training subsets is more effective than averaging over initial weights, or averaging over regularization parameters. This suggests that there is larger independence in errors of nets trained on different small subsets of the high-dimensional data.

### 6.3 Comparison of constraints

**PCA models** Principal component analysis is a standard statistical technique for dimensionality reduction based on the eigenvectors of the data covariance matrix (Duda and Hart, 1973). In the context of classification, this method was used in (Kirby and Sirovich, 1990) for faces and in (Murase and Nayar, 1993) for man-made objects. Later, it was shown that this technique is not optimal (under a given compression rate) for classification (O'Toole et al., 1993; Turk and Pentland, 1993). Our results once again confirm this statement, PCA networks (networks with a regularization value $\lambda = 1$) and their ensembles are inferior to all other models with the same architectural complexity.

**Unconstrained models** Unconstrained (Uncons.) models corresponding to training without reconstruction and WD constraints serve as a base-line for comparison. Unconstrained models are significantly superior to PCA models. This indicates that the features required for classification and reconstruction are not identical (see also Figure 3). Our results show that unconstrained models are inferior to constrained models, mainly when images are degraded.

**Constrained models** We have considered three types of constraints for Pentland-data: WD constraints, reconstruction constraints "Reco." and a combination of both "Reco. + WD" (described in detail in Section 3). It appears, that a WD network chosen via cross-validation is superior to an unconstrained network for all data-degradations. A network with reconstruction constraints may be inferior to the unconstrained network (Table 3, columns marked by "Nets" in mini-tables). Networks with hybrid constraints "Reco. + WD" are superior to WD nets in most of the degradation cases; The more significant improvement is achieved for rotation, illumination, partial occlusion and high-pass filtering.

[3] Note that the test data on with the networks are tested was not used in training or regularizing any of the cross-folds.

Percent misclassification errors and standard deviations (Pentland data-set)

| **Models** | **Clean data** | | | | **Illumination** $\phi = 4.5^o \pm 4.5^o$ | | | |
|---|---|---|---|---|---|---|---|---|
| | Nets | Ens. A | Ens. B | Ens. C | Nets | Ens. A | Ens. B | Ens. C |
| PCA | 12.4/1.6 | - | 10.7/0.9 | 9.7/0.3 | 12.7/1.2 | - | 11.4/1.2 | 10/0.3 |
| Uncons. | 2.4/0.3 | - | 2.1/0.1 | 1.7/0 | 3.2/0.3 | - | 2.6/0.0.3 | 1.9/0.1 |
| WD | 2.2/0.3 | 2.1/0.2 | $\boxed{1.8/0.4}$ | $\boxed{\mathbf{1.4/0.1}}$ | $\boxed{2.3/0.2}$ | $\boxed{2.3/0.3}$ | $\boxed{1.9/0.2}$ | $\boxed{\mathbf{1.2/0.1}}$ |
| Reco. | 2.1/0.3 | 2.1/0.2 | 2.1/0.2 | 1.8/0.1 | 2.8/0.2 | 2.9/0.4 | 2.3/0.3 | 1.8/0.2 |
| Reco. + WD | $\boxed{2.0/0.4}$ | $\boxed{2.0/0.3}$ | 1.9/0.3 | 1.7/0.2 | 2.3/0.3 | 2.4/0.3 | 2.1/0.3 | 1.6/0.1 |

| **Models** | **Rotation** $\alpha = 5^o$ | | | | **Partial occlusion** $k = 3$ | | | |
|---|---|---|---|---|---|---|---|---|
| | Nets | Ens. A | Ens. B | Ens. C | Nets | Ens. A | Ens. B | Ens. C |
| PCA | 14.8/1.1 | - | 13.7/1.8 | 12.6/0.3 | 32.8/0.6 | - | 27.6/0.6 | 26.8/0.7 |
| Uncons. | 4.7/0.4 | - | 4.1/0.2 | 3.2/0.2 | 23.2/1.6 | - | 23.1/0.5 | 21.9/0.7 |
| WD | 3.4/0.5 | $\boxed{3.1/0.2}$ | $\boxed{3.1/0.4}$ | 2.0/0.2 | $\boxed{19.0/1.0}$ | $\boxed{18.0/1.0}$ | $\boxed{17.3/1.0}$ | $\boxed{\mathbf{16.8/0.7}}$ |
| Reco. | 4.4/0 | 3.6/0.2 | 3.9/0.3 | 3.0/0.3 | 22.3/1.1 | 20.7/1.1 | 21.7/1.6 | 21.1/0.6 |
| Reco. + WD | $\boxed{3.1/0.4}$ | 3.1/0.4 | 3.3/0.2 | $\boxed{\mathbf{1.9/0.1}}$ | 20.2/0.9 | 18.2/0.7 | 18.4/0.9 | 17.4/0.4 |

| **Models** | **Gaussian blur** $\sigma = 2$ | | | | **DOG filter** $\sigma_1 = 1,\ \sigma_2 = 3$ | | | |
|---|---|---|---|---|---|---|---|---|
| | Nets | Ens. A | Ens. B | Ens. C | Nets | Ens. A | Ens. B | Ens. C |
| PCA | 18.7/1.4 | - | 15.8/1.6 | 13.9/0.3 | 21.8/1.7 | - | 17.7/1.3 | 14.4/0.6 |
| Uncons. | 8.2/0.5 | - | 7.4/0.7 | 6.6/0.4 | 6.2/0.7 | - | 4.4/0.3 | 3.7/0.3 |
| WD | $\boxed{6.7/0.8}$ | 7.0/0.7 | $\boxed{6.7/0.5}$ | $\boxed{\mathbf{5.3/0.3}}$ | 4.8/0.3 | 4.8/0.5 | $\boxed{3.8/0.4}$ | 3.1/0.3 |
| Reco. | 9.7/0.8 | 7.6/0.8 | 7.6/0.6 | 7.0/0.3 | 5.1/0.7 | 4.6/0.5 | 3.9/0.2 | $\boxed{\mathbf{2.9/0.3}}$ |
| Reco. + WD | 8.6/1.0 | $\boxed{6.9/0.6}$ | 7.3/0.5 | 5.8/0.4 | $\boxed{3.3/0.3}$ | $\boxed{3.6/0.3}$ | 3.9/0.3 | $\boxed{\mathbf{2.9/0.3}}$ |

| **Models** | **Out-of-focus blur** $R = 2$ | | | | **High pass filter** $R = 2$ | | | |
|---|---|---|---|---|---|---|---|---|
| | Nets | Ens. A | Ens. B | Ens. C | Nets | Ens. A | Ens. B | Ens. C |
| PCA | 15.3/1.2 | - | 12.0/1.4 | 10.4/0.2 | 34.6/2.4 | - | 31.7/1.7 | 29.0/0.4 |
| Uncons. | 3.6/0.4 | - | 3.2/0.3 | 2.3/0.2 | 25.6/1.1 | - | 23.2/1.0 | 21/0.8 |
| WD | 3.2/0.4 | 3.0/0.1 | 2.9/0.2 | 2.3/0.2 | 23/1.0 | 22.4/0.7 | 21/0.7 | 21.8/0.5 |
| Reco. | 3.9/0.2 | 4.0/0.3 | 2.9/0.2 | 2.3/0.1 | 22.6/1.1 | 21.3/0.8 | 21.2/1.0 | $\boxed{\mathbf{19.3/0.7}}$ |
| Reco. + WD | $\boxed{2.9/0.2}$ | $\boxed{2.9/0.2}$ | $\boxed{2.8/0.2}$ | $\boxed{\mathbf{2.2/0}}$ | $\boxed{20.1/0.7}$ | $\boxed{21/1.0}$ | $\boxed{20.8/0.9}$ | 20.3/0.5 |

| **Models** | **Gaussian noise** $snr = 1$ | | | | **Salt and pepper noise** $d = 0.2$ | | | |
|---|---|---|---|---|---|---|---|---|
| | Nets | Ens. A | Ens. B | Ens. C | Nets | Ens. A | Ens. B | Ens. C |
| PCA | 13.9/1.9 | - | 12.8/1.5 | 10.7/0.3 | 21.8/1.4 | - | 16.8/0.6 | 15.4/0.5 |
| Uncons. | 3.2/0.4 | - | 3.0/0.3 | 2.2/0.3 | 6.3/0.4 | - | 4.6/0.7 | 3.0/0.6 |
| WD | 2.8/0.3 | 2.9/0.4 | 3.1/0.3 | 2.2/0.3 | 5.0/0.8 | 4.6/0.9 | $\boxed{4.2/0.8}$ | $\boxed{\mathbf{2.4/0.3}}$ |
| Reco. | 3.7/0.5 | 3.2/0.5 | 3.1/0.4 | 2.2/0.2 | 6.7/0.7 | 5.1/0.4 | 5.6/0.9 | 4.1/0.3 |
| Reco. + WD | $\boxed{2.7/0.4}$ | $\boxed{2.6/0.4}$ | $\boxed{2.8/0.3}$ | $\boxed{\mathbf{2.1/0.2}}$ | $\boxed{4.6/0.9}$ | $\boxed{4.2/0.5}$ | 4.7/0.8 | 3.4/0.2 |

Table 3: Each mini-table shows classification performances of the models versus a certain type of degradation (indicated in the first row of mini-tables). Classification performance is given by an averaged percent misclassification rate and its standard deviation (PE/SD) (see Appendix for details on (PE/SD) evaluation). Models are obtained considering different networks and their combination to ensembles (columns) when they are regularized by different degree and type of constraints (rows); the best parameters are given in a companion Table 2. The smallest PE per network architecture (column) is enclosed in a box and a corresponding row shows the best type of constraints versus it. The best overall model versus a degradation (mini-table) is enclosed in a box and bolded. In most of the cases, classification performance of network architectures improves from best networks to Ens. C in the same order as table columns. Models marked with PCA are always significantly inferior to other models of the same architecture. The best WD network is always superior to an unconstrained network for all data-degradations, while the best Reco. net is sometimes inferior to an unconstrained network. Networks with combined constraints (Reco. + WD) are superior to WD. nets in 7 degradation cases from 10. In most of the cases, among ensembles of types A (Ens. A) the best are ensembles with combined (Reco. + WD) constraints. For Ens. B-C the best ensembles are with combined (Reco. + WD) or WD constraints depending on a degradation type. Taken as a base-line for comparison unconstrained Ens. C, the most significant improvement is achieved for clean data and such degradations as rotation, illumination, partial occlusion and high-pass filtering.

Percent misclassification errors and standard deviations (TAU data-set)

| **Models** | **Clean data** | | | | **Illumination** $\phi = 4.5^o \pm 4.5^o$ | | | |
|---|---|---|---|---|---|---|---|---|
| | Nets | Ens. A | Ens. B | Ens. C | Nets | Ens. A | Ens. B | Ens. C |
| PCA | 36.5/3.7 | - | 32.8/3.2 | 26.5/0.5 | 47.1/3.2 | - | 41.2/4.3 | 36.8/0.2 |
| Uncons. | 20.5/4.7 | - | 15.1/2.7 | 11.7/1.0 | 30.9/5.0 | - | 27.5/4.3 | 21.8/0.7 |
| WD | 17.8/2.7 | 16.7/3.6 | 15.3/2.6 | 9.8/0.8 | 28.8/4.5 | 28.4/4.9 | 26.8/4.0 | **18.8/0.9** |
| Reco. + WD | 15.3/1.8 | 14.9/2.2 | 14.9/2.1 | **9.0/0.1** | 27.0/4.1 | 26.8/3.7 | 27.0/3.6 | 19.4/0.2 |

| **Models** | **Rotation** $\alpha = 5^o$ | | | | **Partial occlusion** $k = 1.5$ | | | |
|---|---|---|---|---|---|---|---|---|
| | Nets | Ens. A | Ens. B | Ens. C | Nets | Ens. A | Ens. B | Ens. C |
| PCA | 46.9/1.0 | - | 41.7/1.1 | 37.0/0.7 | 43.2/3.5 | - | 36.5/2.4 | 31.4/0.4 |
| Uncons. | 32.4/4.6 | - | 27.0/3.8 | 23.7/0.9 | 25.9/3.8 | - | 21.0/2.6 | 15.4/0.4 |
| WD | 27.5/2.9 | 28.4/3.4 | 25.7/2.7 | 20.4/0.9 | 23.4/3.2 | 22.3/2.5 | 20.5/2.6 | 15.4/0.6 |
| Reco. + WD | 26.6/2.8 | 27.0/3.5 | 24.1/2.3 | **19.4/0.9** | 21.0/2.9 | 19.6/2.2 | 18.7/1.6 | **12.5/0.8** |

| **Models** | **Gaussian blur** $\sigma = 2$ | | | | **DOG filter** $\sigma_1 = 1,\ \sigma_2 = 2$ | | | |
|---|---|---|---|---|---|---|---|---|
| | Nets | Ens. A | Ens. B | Ens. C | Nets | Ens. A | Ens. B | Ens. C |
| PCA | 47.1/2.2 | - | 42.8/2.7 | 37.7/0.9 | 46.4/2.9 | - | 44.4/2.1 | 33.5/1.3 |
| Uncons. | 32.7/4.6 | - | 29.3/3.0 | 25.5/0.3 | 25.0/3.4 | - | 21.0/3.0 | 15.2/1.5 |
| WD | 30.9/3.2 | 30.0/3.8 | 28.1/2.4 | **22.6/1.4** | 23.0/2.5 | 22.5/3.0 | 19.8/3.3 | 13.9/0.8 |
| Reco. + WD | 27.7/2.6 | 28.2/3.0 | 27.5/3.3 | 23.5/0.3 | 22.3/2.9 | 20.1/2.1 | 19.4/2.6 | **12.8/1.7** |

| **Models** | **Out-of-focus blur** $R = 2$ | | | | **High pass filter** $R = 2$ | | | |
|---|---|---|---|---|---|---|---|---|
| | Nets | Ens. A | Ens. B | Ens. C | Nets | Ens. A | Ens. B | Ens. C |
| PCA | 36.5/3.2 | - | 34.0/2.7 | 28.0/0.9 | 50.7/2.1 | - | 42.8/2.0 | 34.5/0.9 |
| Uncons. | 23.0/4.5 | - | 18.2/3.4 | 13.0/2.1 | 26.8/2.9 | - | 22.3/2.2 | 17.6/1.6 |
| WD | 20.5/2.8 | 20.1/3.6 | 16.9/2.6 | 12.0/1.0 | 24.8/1.6 | 24.6/1.3 | 22.1/1.5 | 15.7/1.3 |
| Reco. + WD | 18.7/1.8 | 18.2/1.9 | 16.7/1.8 | **10.6/0.3** | 23.2/2.1 | 20.1/1.6 | 20.3/1.4 | **13.9/0.3** |

| **Models** | **Gaussian noise** $snr = 10$ | | | | **Salt and pepper noise** $d = 0.2$ | | | |
|---|---|---|---|---|---|---|---|---|
| | Nets | Ens. A | Ens. B | Ens. C | Nets | Ens. A | Ens. B | Ens. C |
| PCA | 36.5/3.5 | - | 33.1/3.6 | 26.4/0.4 | 53.4/4.8 | - | 45.1/2.6 | 35.6/0.4 |
| Uncons. | 21.4/4.3 | - | 15.8/2.8 | 11.3/0.4 | 36.9/4.1 | - | 27.7/3.6 | 23.8/0.8 |
| WD | 18.2/2.8 | 16.2/3.4 | 15.1/3.0 | 10.6/0.8 | 31.1/3.6 | 29.3/3.8 | 27.7/3.6 | 22.1/0.7 |
| Reco. + WD | 16.2/1.9 | 15.5/2.4 | 14.6/2.3 | **9.3/0.7** | 27.3/4.1 | 27.3/3.8 | 27.0/5.0 | **20.4/1.0** |

Table 4: This table and all notations are the same as in Table 3; models parameters are given in Table 2. In most of the cases (similar to Pentland data-set), classification performance of network architectures improves from best networks to Ens. C in the same order as table columns. It is impressive that Ens. C are better than Ens. A–B despite the less number on networks composing Ens. C (3 nets versus 4 in Ens. A–B ). In about all the cases combined "Reco. + WD" constraints are better than WD constraints for all network architectures. Network architectures with WD constraints are better than unconstrained models and PCA models are always significantly inferior to other models of the same architecture. It is clearly seen, that regularized network architectures not only have smaller percent error rates but also have smaller deviations.

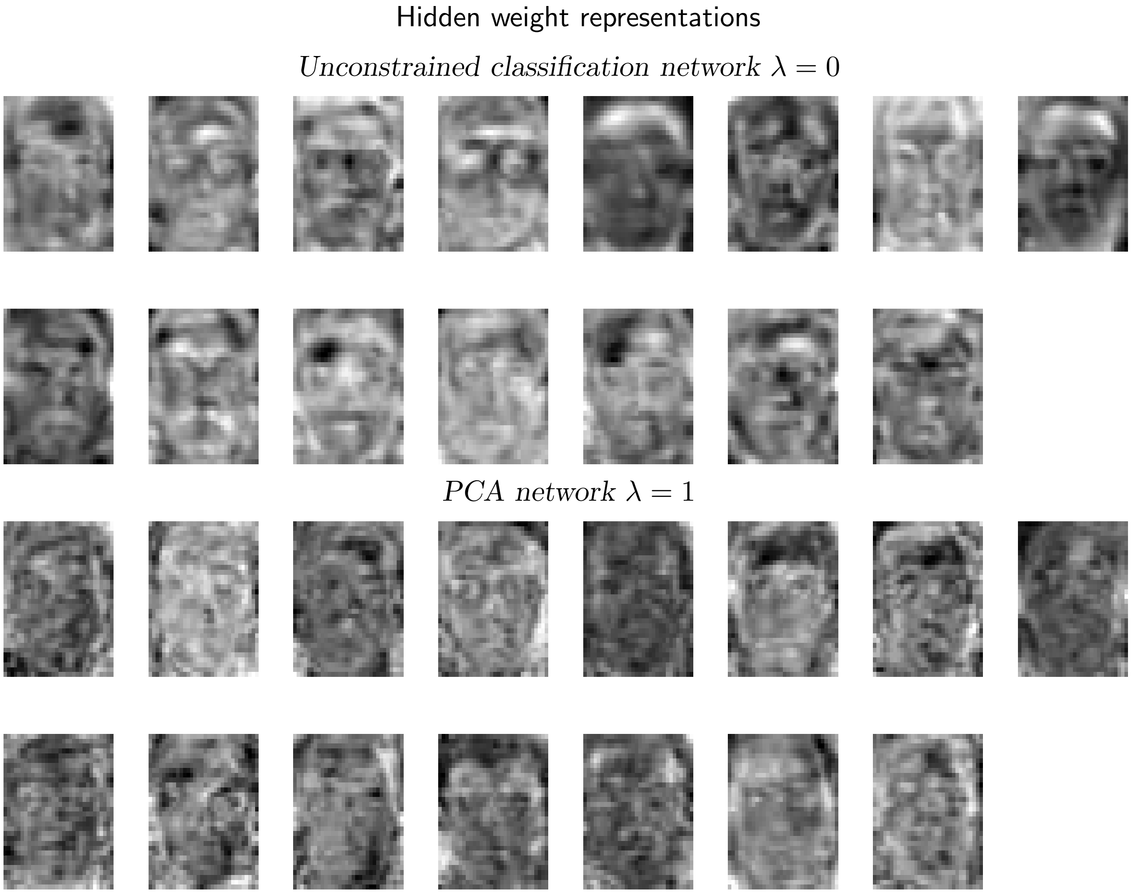
Hidden weight representations
Unconstrained classification network $\lambda = 0$
PCA network $\lambda = 1$

Figure 3: Hidden weights presented as images. There is an evident difference between PCA and classification constraints.

For TAU data-set, only two types of constraints were considered: WD and its combination with reconstruction constraints “Reco. + WD”. The results (Table 4) are easier to interpret since “Reco. + WD” constraints are better than WD constraints for all network architectures. In addition, regularized network architectures not only have smaller error rates but also have smaller deviations.

**Hidden representation of constrained networks** It was already shown (Figure 3) that hidden representations of unconstrained and PCA networks are not the same. Hidden weights of classification networks as image filters look more informative and contrast in the areas of eyes, mouth and hairs than hidden weights of PCA networks. At the same time PCA networks being asymptotically equivalent to a standard PCA method extract hidden weights that are about orthogonal that may be a desired property for efficient data coding. In order to test an orthogonality property of a single network a following orthogonality measure $R$ was introduced:

$$R = 1 - \frac{1}{h(h-1)} \sum_{i=1}^{h-1} \sum_{j>i}^{h} |w_j \cdot w_i|, \tag{10}$$

where $w_i, i = 1 \dots h$ is a vector of hidden weights connecting a hidden unit $i$ with the input layer. This measure is nonnegative and is equal to its maximum value 1 when all hidden weights are mutually orthogonal. Since in our experiments each network with the specific $\lambda$ value is presented by a committee of networks trained on different data to get a stable result LHS. of Eq. 10 is averaged and its standard deviation is found similar to CV approach for PE and SD (see Appendix). An orthogonality measure of hidden weights of the networks with different constraints is presented in Figure 4 and shows that as a degree of reconstruction constraints grows an orthogonality measure of hidden weights for constrained networks grows as well. Regularization with WD constraints also orthogonalizes hidden weights in the beginning but starting from some value orthogonality measure falls down.

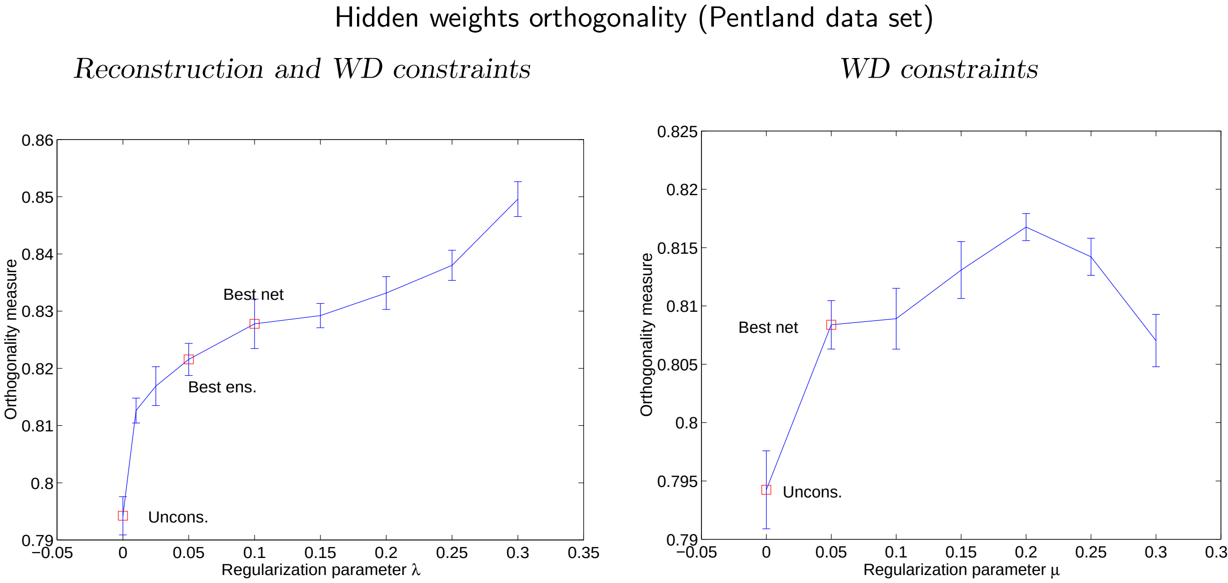

Figure 4: When the degree of reconstruction constraints grows, the orthogonality between the hidden weights. For WD constraints, orthogonality grows at the beginning and then goes down again. For PCA networks ($\lambda = 1$) an orthogonality measure $R = 0.95$ and its std is 0.01; (it is out of its limits on the left panel).

This finding allows us to conceive reconstruction constraints via a minimum description length principle (MDL) (Rissanen, 1985). Following the MDL principle for a supervised task (Hinton and Zemel, 1997) one searches for a model that allows to encode the input data efficiently and to reduce model prediction errors of the output simultaneously (classification output in our consideration). Reconstruction constraints regularizing hidden representation of the classification/reconstruction network lead to more efficient input data encoding. Indeed, principal components are uncorrelated and principal eigenvectors are orthogonal, while hidden weights of unconstrained classification networks may have a large correlation. As we have seen one of the roles of reconstruction constraints is in hidden units decorrelation.

## 7 Conclusion

We have studied a real-world classification problem under realistic degradation conditions.

In particular, we have studied the case where the distribution of the training data does not match the distribution of the test data, due to different image degradations. This is a very challenging task that is unfortunately much less studied. For an overview of existing methods that address degraded image classification see (Stainvas and Intrator, 2000).

We have compared the effects of ensembles based on different variables: initial conditions, regularization parameter and cross-folds. We find that averaging over cross-folds is far superior to averaging over other variables, even at the price of training on smaller size sub-samples.

We have studied reconstruction constraints as a regularization method for classification schemes. Our finding here is a bit surprising; We find that a careful search for optimal weight decay parameters leads to comparable and possibly superior performance to networks that were trained only with reconstruction constraints. However, the combination of both constraints, can further improve the performance. We have also demonstrated that an optimal degree of regularization for ensembles is different than the optimal value for single networks. This results from the variance reduction property of ensembles while keeping the bias portion of the error unchanged, thus implying that networks trained for ensemble should have a lower bias (since the variance reduction is achieved by esembling) (Raviv and Intrator, 1996; Naftaly et al., 1997).

# Appendix: Technical details of model evaluation

We denote networks with trained on different cross-folds and with different initial conditions by by $\mathcal{N}_\lambda(w_j, cf_k)$ and their classification outputs on the data X by $n_\lambda(w_j, cf_k; X)$, $j = 1 \ldots J$, $k = 1 \ldots K$. There are two approaches to treat regularized models (Moody, 1991; Taniguchi and Tresp, 1997): (i) when each unconstrained network with $\lambda = 0$ trained from certain initial weights is considered as a distinct model corresponding to some local minimum and requiring a particular degree of regularization; in this case an optimal regularization parameter depends on initial weights $\lambda^\star(w_j)$; (ii) when the same regularization parameter is found for different initial weights. We tried both approaches and found the first one to be superior in our case. Cross-validation was used to asses regularization effects:

$$PE(\lambda, w) = \frac{1}{K} \sum_1^K \mathcal{E}r(n_\lambda(w, cf_k; D_k)) \tag{11}$$

$$SD^2(\lambda, w) = \frac{1}{K} \sum_1^K (\mathcal{E}r(n_\lambda(w, cf_k; D_k)) - PE(\lambda, w))^2 / K \tag{12}$$

where $\mathcal{E}r(n_\lambda(w, cf_k; D_k))$ is the misclassification rate of the network $\mathcal{N}_\lambda(w, cf_k)$ on the validation data $D_k$. An optimal parameter $\lambda^\star(w)$ is found by minimizing (11). Finally, we evaluate $PE(\lambda^\star, w)$ and $SD(\lambda^\star, w)$ on an unseen *test* data (T), by substituting the unseen data instead of $D_k$ in (11, 12). This estimation of SD's assumes that errors of networks trained on different cross-folds of the same size have similar distribution.

**Ens. A averaged over $\lambda$** We combine networks trained on the same data $cf_k$ and with the same initial weights $w$ and with several "good" $\lambda$ values into a simple regression ensembles:

$$e(w, cf_k; X) = \frac{1}{I} \sum_{i=1}^{I} n_{\lambda_i}(w, cf_k; X)$$

The number of networks in ensembles A is $I$. Percent error rate and its deviation for Ens. A is estimated similar to (11, 12) by replacing $n_\lambda(w, cf_k; X)$ with $e(w, cf_k; X)$. In other words, cross-validation approach is used to estimate ensemble performances. Since networks are averaged over

$\lambda$ the problem of model selection is simplified. It is easier to find an effective region of parameters than to pick up one unique "good" value versus a large data variability.

**Ens. B averaged over initial weights** Ensembles B are composed of networks that were trained on the same cross-fold with a fixed regularization value but with different initial conditions.

$$e(\lambda, cf_k; X) = \frac{1}{J} \sum_{j=1}^{J} n_\lambda(w_j, cf_k; X)$$

This ensemble is a regularized ensemble of networks depended on $\lambda$. An estimation of $\lambda$ parameters, percent error rates and their SD's is the same as for single nets (11, 12) with replacing $n(\lambda, cf_k; X)$ by $e(\lambda, cf_k; X)$.

**Ens. C averaged over training data** Ensembles C are composed of networks that were trained with the same initial weights and with a fixed regularization value but with different training cross-folds.

$$e(\lambda, w; X) = \frac{1}{K} \sum_{k=1}^{K} n_\lambda(w, cf_k; X)$$

This ensemble is a regularized ensemble of networks depended on $\lambda$. To estimate PE and their SD's we use formulae similar to (11, 12) but with averaging over initial weight conditions:

$$\begin{aligned} PE(\lambda) &= \frac{1}{J} \sum_{1}^{J} \mathcal{E}r(e(\lambda, w_j; T)) \\ SD^2(\lambda) &= \frac{1}{J} \sum_{1}^{K} (\mathcal{E}r(e(\lambda, w_j; T)) - PE(\lambda))^2 / J \end{aligned}$$

The number of networks in bagging ensemble (Ens. C) is equal to the number of cross-folds $K$. For bagging ensemble we don't select a optimal $\lambda$ parameter but instead take it to be the same as for Ens. B. For comparison between ensembles of types A–C to be fair the number of networks in ensembles is taken to be the same, if it is not emphasized additionally.

### Acknowledgments

The Computer Vision group in Tel-Aviv University, especially Prof. Hezy Yeshurun, Ariel Tankus and Danny Reisfeld, were very helpful in our many fruitful discussions. They also provided the TAU data and the preprocessing software. Inna Stainvas is very grateful to Felix Abramovich for a fruitful discussion of regularization problems in statistics.

## References

Baluja, S. and Pomerleau, D. A. (1995). Using the representation in a neural network's hidden layer for task-specific focus of attention. In *Proceedings of the International Joint Conference on Artificial Intelligence*, Montreal, Canada.

Bellman, R. E. (1961). *Adaptive Control Processes.* Princeton University Press, Princeton, NJ.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition.* Oxford University Press.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees.* The Wadsworth Statistics/Probability Series, Belmont, CA.

Cannon, M. (1976). Blind deconvolution of spatially invariant image blurs with phase. *icassp*, 24:58–63.

Caruana, R. (1993). Multitask connectionist learning. In *Proceedings of the 1993 Connectionist Models Summer School*, pages 372–379, San Mateo, CA.

Caruana, R. and de Sa, V. R. (1997). Promoting poor features to supervisors: Some inputs work better as outputs. In Mozer, M. C., Jordan, M. I., and Petsche, T., editors, *Advances in Neural Information Processing Systems*, volume 9. Morgan Kaufmann, San Mateo, CA.

Carven, P. and Wahba, G. (1979). Smoothing noisy data with spline functions: estimationg the correct degree of smoothing by the method of generalized cross-validation. *Numer. Math.*, 31:377–403.

Duda, R. O. and Hart, P. E. (1973). *Pattern Classification and Scene Analysis.* John Wiley, New York.

Geman, S., Bienenstock, E., and Doursat, R. (1992). Neural networks and the bias-variance dilemma. *Neural Computation*, 4:1–58.

Gluck, M. A. and Myers, C. E. (1993). Hippocampal mediation of stimulus representation: A computational theory. *Hippocampus*, 3(4):491–516.

Hansen, L. K. and Salamon, P. (1990). Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10):993–1001.

Hastie, T. and Tibshirani, R. (1990). *Generalized Additive Models.* Chapman and Hall, London.

Heskes, T. (1998). Selecting weighing factors in logarithmic opinion pools. *Advances in Neural Information Processing Systems.*

Hinton, G. E. and Zemel, R. S. (1997). Minimizing description length in an unsupervised neural network. Preprint.

Intrator, N. and Cooper, L. N. (1992). Objective function formulation of the BCM theory of visual cortical plasticity: Statistical connections, stability conditions. *Neural Networks*, 5:3–17.

Intrator, N. and Edelman, S. (1996). Making a low-dimensional representation suitable for diverse tasks. *Connection Science, Special issue on Reuse of Neural Networks Through Transfer*, 8(2):205–224. Also in *Learning to Learn*, S. Thrun and L. Pratt ed., Kluwer press.

Intrator, N., Reisfeld, D., and Yeshurun, Y. (1996). Face recognition using a hybrid supervised/unsupervised neural network. *Pattern Recognition Letters*, 17:67–76.

Kandel, E. R. and Schwartz, J. H. (1991). *Principles of Neural Science.* Elsevier, New York, third edition.

Kirby, M. and Sirovich, L. (1990). Application of the Karhunen-Loève procedure for characterization of human faces. *PAMI*, 12(1):103–108.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. *International Joint Conference on Artificial Intelligence.*

Kramer, M. A. (1991). Nonlinear principal component analysis using autoassociative neural networks. *AIChe Journal*, 37(2):233–243.

Krogh, A. and Vedelsby, J. (1995). Neural network ensembles, cross validation, and active learning. In Tesauro, G., Touretzky, D., and Leen, T., editors, *Advances in Neural Information Processing Systems*, volume 7, pages 231–238, Cambridge, MA. MIT Press.

Lai, S.-H. and Fang, M. (1999). Robust and efficient image alignment with spatially varying illumination models. In *Proceedings of the Computer Vision and Pattern Recognition*, volume 2, Institute of Electrical and Electronics Engineers.

Leung, H. C. and Zue, V. W. (1989). Applications of error back-propagation to phonetic classification. In Touretzky, D. S., editor, *Advances in Neural Information Processing Systems 1*, volume 1, pages 206–214. Morgan Kaufmann, San Mateo, CA.

Luenberger, D. G. (1984). *Linear and Nonlinear Programming.* Addison-Wesley Inc, Massachusetts. second edition.

MacKay, D. J. C. (1992). A practical Bayesian framework for backprop networks. *Neural Computation*, 4:448–472.

Marr, D. (1982). *Vision.* Imprint FREEMAN, New York.

Moody, J. E. (1991). Note on generalization, regularization and architecture selection in nonlinear learning systems. In Juang, B. H., Kung, S. Y., and Kamm, C. A., editors, *Neural Networks for Signal Processing – Proceedings of the 1991 IEEE Workshop*, pages 1–10.

Murase, H. and Nayar, S. K. (1993). Learning object models from appearance. *Proceedings of the Eleventh National Conference on Artificial Intelligence.*

Naftaly, U., Intrator, N., and Horn, D. (1997). Optimal ensemble averaging of neural networks. *Network*, 8(3):283–296.

Oja, E. (1995). PCA, ICA, and nonlinear hebbian learning. In *Proc. Int. Conf. on Artificial NeuralNetworks ICANN-95*, pages 89–94.

O'Toole, A. J., Valentin, D., and Abdi, H. (1993). A low dimensional representation of faces in the higher dimensions of the space. *Journal of the Optical Society of America, series A*, 10:405–411.

Poggio, T. and Girosi, F. (1994). A theory of networks for approximation and learning. *A.I. Memo No.1140,C.B.I.P. Paper No.31, Massachusetts Institute of technology.*

Pomerleau, D. A. (1993). Input reconstruction reliablility estimation. In Giles, C. L., Hanson, S. J., and Cowan, J. D., editors, *Advances in Neural Information Processing Systems*, volume 5, pages 279–286. Morgan Kaufmann.

Raviv, Y. and Intrator, N. (1996). Bootstrapping with noise: An effective regularization technique. *Connection Science, Special issue on Combining Estimators*, 8:356–372.

Ripley, B. D. (1996). *Pattern Recognition and Neural Networks.* Oxford Press.

Rissanen, J. (1985). Minimum description length principle. *Encyclopedia of Statistical Sciences*, pages 523–527.

Rognvaldsson, T. S. (1998). A simple trick for estimating the weight decay parameter. In Orr, G. B. and Müller, K., editors, *Neural Networks: Tricks of the Trade*, pages 71–93. Springer.

Rosenfeld, A. and Kak, A. C. (1982). *Digital Picture Processing.* Academic press, New York.

Stainvas, I. and Intrator, N. (2000). Blurred face recognition via a hybrid network architecture. In *Proceedings Int. Conf. on Pattern Recognition*, volume 2, pages 809–812.

Taniguchi, M. and Tresp, V. (1997). Averaging regularized estimators. *Neural Computation*, 9:1163–1178.

Tankus, A. (1996). *Automatic face detection and recognition.* Master thesis, Tel-Aviv University.

Tankus, A., Yeshurun, Y., and Intrator, N. (1997). Face detection by direct convexity estimation. *Pattern Recognition Letters*, 18(9):913–922.

Turk, M. and Pentland, A. (1993). Experiments with eigenfaces. *Looking At People Workshop, IJCAI'93*, pages 1–6.

Wahba, G. (1990). *Splines Models for Observational Data.* Series in Applied Mathematics, Vol. 59, SIAM, Philadelphia.

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5:241–259.

Yang, H. and Amari, S. (1997). Adaptive on-line learning algorithms for blind separation – maximum entropy and minimum mutual information. *Neural Computation*, 9(7):1457–1482.