

# Neural Networks with Marginalized Corrupted Hidden Layer

YanJun Li<sup>1</sup>, Xin Xin<sup>1(✉)</sup>, and Ping Guo<sup>1,2</sup>

<sup>1</sup> School of Computer Science and Technology, Beijing Institute of Technology,  
Beijing 100081, China

liyanjunzgz@126.com, xxin@bit.edu.cn, pguo@ieee.org

<sup>2</sup> Image Processing and Pattern Recognition Laboratory,  
Beijing Normal University, Beijing 100875, China

**Abstract.** Overfitting is an important problem in neural networks (NNs) training. When the number of samples in the training set is limited, explicitly extending the training set with artificially generated samples is an effective solution. However, this method has the problem of high computational costs. In this paper we propose a new learning scheme to train single-hidden layer feedforward neural networks (SLFNs) with implicitly extended training set. The training set is extended by corrupting the hidden layer outputs of training samples with noise from exponential family distribution. When the number of corruption approaches infinity, in objective function explicitly generated samples can be expressed as the form of expectation. Our method, called marginalized corrupted hidden layer (MCHL), trains SLFNs by minimizing the loss function expected values under the corrupting distribution. In this way MCHL is trained with infinite samples. Experimental results on multiple data sets show that MCHL can be trained efficiently, and generalizes better to test data.

**Keywords:** Neural network · Overfitting · Classification

## 1 Introduction

Overfitting is an important problem in NNs training [1,2]. Overfitting occurs when the network has too many free parameters relative to the number of training samples. In this situation the network adapts to the particular details of the training set and leads to poor generalization performance.

A typical method of solving overfitting is to extend the training set. Because overfitting can become less severe as the size of the training set increases. When the number of samples in the training set is limited, explicitly extending the training set with artificially generated samples is an effective solution. However, this method has the problem of high computational costs. Because the training time increases with the number of samples.

To solve overfitting, in this paper we propose a new learning scheme to train single-hidden layer feedforward neural networks (SLFNs) with implicitly

extended training set. The training set is extended by corrupting the hidden layer outputs of training samples with noise from exponential family distribution. When the number of corruption approaches infinity, in objective function explicitly generated samples can be expressed as the form of expectation. Our method, called marginalized corrupted hidden layer (MCHL), trains SLFNs by minimizing the loss function expected values under the corrupting distribution. In this way MCHL is trained with infinite samples.

Parameter optimization of NNs is a big challenge. NNs are normally optimized with backpropagation (BP) algorithm [3]. As a first order gradient descent parameter optimization method, BP algorithm has the problems of local minimum and slow convergence.

To optimize MCHL efficiently, we propose to optimize the parameters of MCHL by pseudo inverse operation. Our optimization method is inspired by the work of Guo and Michael [4]. Different from BP algorithm, for MCHL the previously trained weights in the network are not changed. This makes the training of MCHL more efficient. In addition, the model parameters have analytical solution, so MCHL tends to achieve global minima.

Experimental results on multiple data sets show that MCHL can be trained efficiently, and generalizes better to test data. In summary, we make the following contributions: (1) to solve overfitting, we propose a new learning scheme to train SLFNs with implicitly extended training set; (2) for MCHL, we propose an efficient parameter optimization method; (3) on several data sets, we show that MCHL can be trained efficiently, and generalizes better to test data.

## 2 Related Works

In this subsection we brief the related works about corrupting samples during training. Burges and Scholkopf [5] first propose to improve the generalization ability of predictors by explicitly corrupting training data. Hinton et al. [1] propose a method to reduce overfitting by randomly omitting half of the feature detectors on each training case. Vincent et al. [6] propose a unsupervised representation learning method, which corrupts the input data and keeps the desired output unchanged. Their approach is commonly used to train autoencoders, and these denoising autoencoders can be further used to initialize deep architectures. Chen et al. [7] propose marginalized denoising autoencoders for domain adaptation which are linear denoising autoencoders. Maaten et al. [8] propose to extend the training set with infinitely many artificial samples by corrupting the original training data. Our method differs from Maaten et al. work in that we corrupt the hidden layer outputs instead of the original training samples. Except corrupting features, there is another research direction (corrupting labels). Chen et al. [9] propose a fast image annotation method based on labels corruption. Lawrence and Schölkopf [10] propose an algorithm for constructing a kernel Fisher discriminant from training examples with noisy labels.

### 3 MCHL

In this section we first introduce MCHL learning scheme, then analyze how to marginalize the noise introduced in the hidden layer outputs analytically by minimizing the loss function expected values under the corrupting distribution, i.e., solve the weights of hidden layer to output layer.

#### 3.1 Learning Scheme

Given a training set  $D = \{(\mathbf{x}_i, \mathbf{y}_i) | \mathbf{x}_i \in \mathbf{R}^d, \mathbf{y}_i \in \mathbf{R}^k, i = 1, \dots, M\}$ , let  $L$  denotes the number of hidden nodes,  $h(\mathbf{x})$  denotes the feature mapping function and  $\mathbf{h}(\mathbf{x})$  denotes mapping result of data  $\mathbf{x}$ . In MCHL different hidden neurons can use different feature mapping functions. In real applications  $h(\mathbf{x})$  can be defined as

$$h(\mathbf{x}) = F(\mathbf{a}, b, \mathbf{x}), \mathbf{a} \in \mathbf{R}^d, b \in R, \quad (1)$$

where  $(\mathbf{a}, b)$  are hidden node parameters.  $F(\mathbf{a}, b, \mathbf{x})$  can be any piecewise continuous function which meets universal approximation capability theorem [11]. Typically used feature mapping functions  $F(\mathbf{a}, b, \mathbf{x})$  include Sigmoid ( $\frac{1}{1+\exp(-(\mathbf{a} \cdot \mathbf{x} + b))}$ ), Gaussian ( $\exp(-b\|\mathbf{x} - \mathbf{a}\|)$ ), Hyperbolic tangent ( $\frac{1-\exp(-(\mathbf{a} \cdot \mathbf{x} + b))}{1+\exp(-(\mathbf{a} \cdot \mathbf{x} + b))}$ ) and Cosine ( $\cos(\mathbf{a} \cdot \mathbf{x} + b)$ ).

We use pseudo inverse operation to learn the parameters of MCHL. MCHL trains a single-hidden layer feedforward neural network (SLFN) by two stages: (1) map training data into a new space (called MCHL space); (2) solve the parameters of hidden layer to output layer in MCHL space. We first introduce how to solve parameters of hidden layer to output layer (weight  $\mathbf{W}_2$ ), then depict how to figure out the parameters of input layer to hidden layer (weight  $\mathbf{W}_1$ , i.e. parameters  $\mathbf{a}$  and  $b$ ).

MCHL solves weight  $\mathbf{W}_2$  of hidden layer to output layer in MCHL feature space by minimizing the training error

$$\min_{\mathbf{w}_2} \|\mathbf{H}\mathbf{W}_2 - \mathbf{Y}\|^2, \quad (2)$$

where  $\mathbf{H}$  is the matrix of hidden layer outputs. The smallest norm least squares solution of optimization problem (2) is  $\mathbf{W}_2^* = \mathbf{H}^\dagger \mathbf{Y}$ , where  $\mathbf{H}^\dagger$  is the pseudo inverse of matrix  $\mathbf{H}$ .

We hope the outputs of hidden layer are irrelevant. To achieve this objective we first randomly generate a  $M \times L$  full rank matrix  $\mathbf{P}$ . Huang et al. [11] have proved that SLFNs with arbitrarily assigned input weights and hidden layer biases and with almost any nonzero activation function can universally approximate any continuous functions on any compact input sets. Solve equation

$$\mathbf{X}\mathbf{W}_1 = \mathbf{P}, \quad (3)$$

we get  $\mathbf{W}_1^* = \mathbf{X}^\dagger \mathbf{P}$ . Different from conventional NNs trained with BP algorithm, parameters in MCHL have analytical solutions, hence MCHL can be trained efficiently.

### 3.2 Marginalizing the Noise

Intuitively we can improve the generalization ability of SLFNs by extending the training set in the MCHL feature space. This can be achieved by explicitly corrupting each training sample in the MCHL feature space.

Given the training set  $D$ , let  $\mathbf{t}_i$  denotes the mapping result of sample  $\mathbf{x}_i$  in MCHL feature space, i.e.  $\mathbf{t}_i = \mathbf{h}(\mathbf{x}_i)$ . We can corrupt each sample in MCHL feature space  $N$  times according to a fixed noise distribution to generate a new data set  $\tilde{D}$  with  $MN$  samples. For each sample  $\mathbf{t}_m$  in MCHL feature space, corruption corresponds to generate new samples  $\tilde{\mathbf{t}}_{mn}$  (with  $n = 1, \dots, N$ ). For convenience, we take binary classification,  $y \in \{-1, +1\}$ , for example. The newly generated data set  $\tilde{D}$  can be used for training by minimizing

$$\mathcal{L}(\tilde{D}; \Theta) = \sum_{m=1}^M \frac{1}{N} \sum_{n=1}^N L(\tilde{\mathbf{t}}_{mn}, y_m; \Theta), \quad (4)$$

where  $\tilde{\mathbf{t}}_{mn} \sim p(\tilde{\mathbf{t}}_{mn}|\mathbf{t}_m)$ ,  $\Theta$  is the set of model parameters, and  $L(\tilde{\mathbf{t}}_{mn}, y_m; \Theta)$  is the loss function of the model. The binary classification can be extended to multiclass (with  $k$  classes) by replace label  $y$  with label vector  $\mathbf{y} = \{-1, 1\}^k$ .

Explicit corruption is effective, but it has the problem of high computational costs. The computational complexity of the minimization of  $\mathcal{L}(\tilde{D}; \Theta)$  scales linearly in the number of corrupted samples. Here, we consider the limiting case, i.e.  $N \rightarrow \infty$ . By applying the weak law of large numbers, we can rewrite  $\frac{1}{N} \sum_{n=1}^N L(\tilde{\mathbf{t}}_{mn}, y_m; \Theta)$  as its expectation, i.e.,

$$\mathcal{L}(\tilde{D}; \Theta) = \sum_{m=1}^M E[L(\tilde{\mathbf{t}}_m, y_m; \Theta)]_{p(\tilde{\mathbf{t}}_m|\mathbf{t}_m)}, \quad (5)$$

We assume: (1) corruption distribution is a member of the natural exponential family and the corruption of each dimension of  $\mathbf{t}$  is independent; (2) corruption distribution is unbiased, that is to say  $E[\tilde{\mathbf{t}}_m]_{p(\tilde{\mathbf{t}}_m|\mathbf{t}_m)} = \mathbf{t}_m$ . Here, we use  $\mathbf{w}$  to denote weights of hidden layer to output layer. When loss function is quadratic loss function,  $\mathbf{w}$  can be achieved by minimizing the loss function:

$$\begin{aligned} \mathcal{L}(\tilde{D}; \mathbf{w}) &= \sum_{m=1}^M E[(\mathbf{w}^T \tilde{\mathbf{t}}_m - y_m)^2]_{p(\tilde{\mathbf{t}}_m|\mathbf{t}_m)} \\ &= \mathbf{w}^T \left( \sum_{m=1}^M E[\tilde{\mathbf{t}}_m] E[\tilde{\mathbf{t}}_m]^T + V[\tilde{\mathbf{t}}_m] \right) \mathbf{w} - 2 \left( \sum_{m=1}^M y_m E[\tilde{\mathbf{t}}_m] \right)^T \mathbf{w} + M \end{aligned} \quad (6)$$

where  $V[\mathbf{t}]$  is the covariance of  $\mathbf{t}$ , and all expectation and covariance are under  $p(\tilde{\mathbf{t}}_m|\mathbf{t}_m)$ . According to assumption (1), we can show that  $V[\mathbf{t}]$  is a diagonal matrix which stores the variances of  $\mathbf{t}$ . Set the derivatives of  $\mathcal{L}(\tilde{D}; \mathbf{w})$  with respect to  $\mathbf{w}$  equal to zero, we obtain the optimal solution

$$\mathbf{w}^* = \left( \sum_{m=1}^M E[\tilde{\mathbf{t}}_m] E[\tilde{\mathbf{t}}_m]^T + V[\tilde{\mathbf{t}}_m] \right)^\dagger \left( \sum_{m=1}^M y_m E[\tilde{\mathbf{t}}_m] \right) \quad (7)$$

Probability density function (PDF), mean and variance of typically used corrupting distributions are listed in Table 1.

**Table 1.** PDF, mean and variance of typically used corrupting distributions.

Noise distribution	PDF	$E[\tilde{t}]_{p(\tilde{t} t)}$	$V[\tilde{t}]_{p(\tilde{t} t)}$
Blankout	$p(\tilde{t} = 0) = q, p(\tilde{t} = \frac{1}{1-q}t) = 1 - q$	$t$	$\frac{q}{1-q}t^2$
Gaussian	$p(\tilde{t} t) = N(\tilde{t} t, \sigma^2)$	$t$	$\sigma^2$
Laplace	$p(\tilde{t} t) = Laplace(\tilde{t} t, \lambda)$	$t$	$2\lambda^2$
Poisson	$p(\tilde{t} t) = Poisson(\tilde{t} t)$	$t$	$t$

In summary, training process of MCHL can be summarized as follow:

Given a training set  $D = \{(\mathbf{x}_i, \mathbf{y}_i) | \mathbf{x}_i \in \mathbf{R}^d, \mathbf{y}_i \in \mathbf{R}^k, i = 1, \dots, M\}$ , feature mapping function  $F(\mathbf{a}, b, \mathbf{x})$ , and hidden neuron number  $L$ ,

step1: Randomly generate a  $M \times L$  full rank matrix  $\mathbf{P}$ .

step2: Evaluate the hidden node parameters  $(\mathbf{a}, b)$  by solving Eq. (3).

step3: Calculate the hidden layer output matrix  $\mathbf{H}$ .

step4: Calculate the output weight  $\mathbf{w}^*$  according to formula (7).

## 4 Experiments

Experiments include three parts: (1) analyze the influence of blankout corruption level  $q$  to classification performance of MCHL (We use blankout noise corruption and assume same noise level  $q$  for each dimension of feature.); (2) analyze the influence of hidden nodes number to classification performance of MCHL; (3) analyze the classification performance of MCHL.

Feature mapping function uses sigmoid function.  $l_2$  regularizer is added to the weights calculation of hidden layer to output layer. Wide type of data sets are used in this section, most of the data sets are taken from UCI Machine Learning Repository [13]. We consider binary classification and multiclass classification two cases. Binary classification data sets include: Colon [12], Diabete [13], SPECTF [13], Heart [14], Madelon [15], Australian Credit [13] and Dimdata [16]. Multiclass classification data sets include: Iris [13], Glass [13], Win [13], Ecoli [13], Segment [13], Vehicle [13] and Letter [13]. The corresponding categories number are 3, 6, 3, 8, 7, 4 and 26, separately. Detailed information about the data sets are listed in Table 2.

### 4.1 Influence of Blankout Corruption Level $q$

We explore the classification performance of MCHL as a function of the blankout corruption level  $q$ . Blankout corruption level  $q = 0$  means that MCHL do not corrupt the hidden layer outputs. We set the hidden nodes number to the sample

**Table 2.** Basic statistics of data sets.

Binary	Colon	Diabete	SPECTF	Heart	Madelon	Australian	Dimdata
# total	62	768	267	270	2600	690	4192
# features	2000	8	44	13	500	6	14
Multiclass	Iris	Glass	Win	Ecoli	Segment	Vehicle	Letter
# total	150	214	178	336	2310	846	20000
# features	4	9	13	7	19	18	16

features number and regularization parameter  $C$  to  $10^{-4}$ . Four data sets are used in this subsection. The training data number for each data set are 150 (Heart), 1400 (Madelon), 40 (Colon) and 150 (SPECTF). Experimental results are listed in Table 3.

**Table 3.** Relation between classification results and blankout corruption level  $q$ .

	q=0	q=0.1	q=0.2	q=0.3	q=0.4	q=0.5	q=0.6	q=0.7	q=0.8	q=0.9
Heart	83.33	<b>85.00</b>	80.83	77.50	70.83	67.50	65.83	63.33	60.00	59.17
SPECTF	80.34	<b>81.20</b>	81.20	81.20	81.20	81.20	81.20	81.20	81.20	81.20
Colon	81.82	81.82	<b>86.36</b>	86.36	86.36	86.36	86.36	86.36	86.36	86.36
Madelon	56.67	57.83	<b>58.50</b>	58.17	58.00	58.00	58.00	57.83	57.83	58.33

From Table 3 we can find that: (1) Marginalizing the noise introduced in the hidden layer outputs can improve the classification results (1.67% on Heart, 0.86% on SPECTF, 4.54% on Colon and 1.83% on Madelon). (2) On SPECTF, Colon and Madelon data sets MCHL consistently improves the classification results on all blankout corruption level; (3) On the whole the best performance tends to be obtained by MCHL with low corruption levels, i.e. the order of  $q$  is around 0.2.

## 4.2 Influence of Hidden Nodes Number

As a kind of SLFNs, hidden layer of MCHL can nonlinear map training data into a high dimensional feature space. In this subsection we analysis the impact of hidden nodes number to the classification performance of MCHL. Heart and Madelon data sets are used in this subsection, and the training data number are 150 and 1400, respectively. Regularization parameter  $C$  is set to  $10^{-4}$ . We analyze two cases,  $q = 0$  (does not have corruption) and  $q = 0.1$  (has corruption). Experimental results are listed in Tables 4 and 5. First column of Tables 5 and 6 corresponds to the primary feature dimension of the data.

From Tables 4 and 5, we can find that an appropriate increase in the number of hidden nodes can improve the classification performance of MCHL. Nonlinear

**Table 4.** Classification accuracies of different hidden nodes number on Heart data set.

	L = 13	L = 50	L = 100	L = 200	L = 400	L = 800	L = 1000	L = 1500	L = 2000
q = 0	78.33	<b>82.50</b>	73.33	70.00	69.17	70.83	72.50	70.83	72.00
q = 0.1	81.67	84.17	83.33	<b>85.00</b>	84.17	83.33	83.33	80.83	83.33

**Table 5.** Classification accuracies of different hidden nodes number on Madelon data set.

	L = 500	L = 50	L = 100	L = 200	L = 400	L = 800	L = 1000	L = 1500	L = 2000
q = 0	55.00	49.00	53.83	55.50	54.33	<b>56.83</b>	55.67	52.67	54.33
q = 0.1	56.50	50.67	52.33	55.17	54.50	58.17	57.67	<b>58.50</b>	58.33

feature mapping in MCHL has a similar effect of kernel function used in support vector machine (SVM).

### 4.3 Classification Performance

This subsection we make detailed experiments to analyze the classification performance of MCHL. SVM is used as baseline. All of data sets are used in the subsection. The training data number for each data set are: 150 (Hear), 1400 (Madelon), 40 (Colon), 150 (SPECTF), 510 (Diabete), 460 (Australian), 1000 (Dimdata), 100 (Iris), 140 (Glass), 120 (Win), 220 (Ecoli), 1540 (Segment), 560 (Vehicle) and 13333 (Letter).

SVM uses popular RBF Kernel ( $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2)$ ). Experimental parameters are selected by cross-validation. Parameters C and  $\gamma$  are searched on grid  $\{2^{-16}, 2^{-14}, 2^{-12}, \dots, 2^{12}, 2^{14}, 2^{16}\}$ . The number of hidden layer nodes is selected on grid  $\{50, 100, 200, 400, 800, 1000, 1500, 2000\}$ . Blankout noise corruption level  $q$  is searched on grid  $\{0, 0.1, 0.2, \dots, 0.9\}$ . Experiments on each dataset are repeated ten times with randomly selected training and test data. The mean and standard deviation of classification accuracy are recorded. Experimental results are shown in Tables 6 and 7. From Tables 6 and 7, we can find that the classification performance of MCHL is slightly better than SVM. In addition the

**Table 6.** Binary class classification performance compare.

Data set	Colon	Diabete	SPECTF	Heart	Madelon	Australian	Dimdata
SVM	<b>89.55</b> ± 5.77	79.22 ± 1.92	<b>82.22</b> ± 3.19	<b>86.33</b> ± 2.01	58.12 ± 1.04	86.43 ± 1.50	95.68 ± 0.24
MCHL	88.64 ± 6.51	<b>79.38</b> ± 1.56	81.54 ± 3.16	85.75 ± 2.75	<b>59.15</b> ± 0.96	<b>87.65</b> ± 1.89	<b>95.74</b> ± 0.29

**Table 7.** Multiclass classification performance compare.

Data set	Iris	Glass	Win	Ecoli	Segment	Vehicle	Letter
SVM	97.80 ± 1.89	<b>68.24</b> ± 4.57	98.10 ± 1.21	87.07 ± 3.27	<b>96.43</b> ± 0.63	<b>84.09</b> ± 1.71	92.85 ± 0.28
MCHL	<b>98.20</b> ± 1.66	67.84 ± 4.79	<b>98.45</b> ± 1.43	<b>87.59</b> ± 2.62	96.14 ± 0.65	83.57 ± 1.57	<b>93.50</b> ± 0.22

parameters of MCHL have analytical solutions, this makes the training efficiency of MCHL higher than SVM.

## 5 Conclusions

Generalization ability of NNs is limited by the number of training samples. Explicitly extending the training set with artificially generated samples by corrupting hidden layer outputs can improve the generalization ability of NNs. But it has the problem of high computation costs. We propose MCHL which improves the generalization ability of SLFNs by marginalizing the noise introduced in the hidden layer outputs. In this way MCHL is trained with infinite samples. Experimental results on multiple data sets show that MCHL can be trained efficiently, and generalizes better to test data.

**Acknowledgments.** The work described in this paper was mainly supported by National Natural Science Foundation of China (No. 61300076 and 61375045), Ph.D. Programs Foundation of Ministry of Education of China (No. 20131101120035), Beijing Natural Science Foundation(4142030), Excellent young scholars Research Fund of Beijing Institute of Technology.

## References

1. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors (2012)
2. Guo, P., Michael, R.L., Chen, C.L.P.: Regularization parameter estimation for feedforward neural networks. *IEEE Trans. Syst. Man Cybern. Part B* **33**(1), 35–44 (2003)
3. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. *Nature* **323**(9), 533–536 (1986)
4. Guo, P., Michael, R.L.: A pseudoinverse learning algorithm for feedforward neural networks with stacked generalization application to software reliability growth data. *Neurocomputing* **56**(1), 101–121 (2004)
5. Burges, C.J.C., Scholkopf, B.: Improving the accuracy and speed of support vector machines. In: *Advances in Neural Information Processing Systems*, vol. 9, pp. 375–381 (1997)
6. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A.: Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.* **11**, 3371–3408 (2010)
7. Chen, M., Xu, Z., Weinberger, K.Q., Sha, F.: Marginalized denoising autoencoders for domain adaptation. In: *International Conference on Machine Learning*, pp. 767–774 (2012)
8. Maaten, L., Chen, M., Tyree, S., et al.: Learning with marginalized corrupted features. In: *International Conference on Machine Learning*, pp. 410–418 (2013)
9. Chen, M., Zheng, A., Weinberger, K.: Fast image tagging. In: *International Conference on Machine Learning*, pp. 1274–1282 (2013)
10. Lawrence, N.D., Schölkopf, B.: Estimating a kernel fisher discriminant in the presence of label noise. In: *International Conference on Machine Learning*, pp. 306–313 (2001)

11. Huang, G.B., Chen, L., Siew, C.K.: Universal approximation using incremental constructive feedforward networks with arbitrary input weights. Technical report ICIS/46/2003
12. Li, J., Liu, H.: Kent Ridge Bio-Medical Data Set Repository (2004). <http://levis.tongji.edu.cn/gzli/data/mirror-kentridge.html>
13. Bache, K., Lichman, M.: UCI Machine Learning Repository (2013). <http://archive.ics.uci.edu/ml>
14. Mike, M.: Statistical Datasets (1989). <http://lib.stat.cmu.edu/datasets/>
15. Guyon, I.: Design of experiments of the NIPS 2003 variable selection benchmark (2003). <http://www.nipsfsc.ecs.soton.ac.uk/papers/Datasets.pdf>
16. Odewahn, S.C., Stockwell, E.B., Pennington, R.L., Humphreys, R.M., Zumach, W.A.: Automated star/galaxy discrimination with neural networks. *Astronomical Journal* **103**(1), 318–331 (1992)