

Submitted to *Neural Networks*.

Dynamics of Batch Learning in Multilayer Networks

– Overrealizability and Overtraining –

Kenji Fukumizu

The Institute of Physical and Chemical Research (RIKEN)

E-mail: fuku@brain.riken.go.jp

December 28, 1998

Abstract

This paper investigates the dynamics of batch learning of multilayer neural networks in the asymptotic case where the number of training data is much larger than the number of parameters. We consider regression problems assuming noisy output data. First, we present experimental results on the behavior in the steepest descent learning of multilayer perceptrons and three-layer linear neural networks. We see in these results that strong overtraining, which is the increase of the generalization error in training, occurs if the model has surplus hidden units to realize the target function. Next, to analyze overtraining from the theoretical viewpoint, we consider the solution of the steepest descent learning equation of a three-layer linear neural network, and prove that a network with surplus hidden units shows overtraining. From this theoretical analysis, we can see that overtraining is not a feature observed in the final stage of learning, but in an intermediate time interval.

Keywords: Overtraining; Overrealizable; Multilayer networks; Linear neural networks

1 Introduction

This paper discusses the dynamics of batch learning in multilayer neural networks. Multilayer networks like multilayer perceptrons have been extensively used in many applications, in hope that their nonlinear structure inspired by biological neural networks shows a variety of advantages. However, there are also some weaknesses caused by the nonlinearity. To train a neural network,

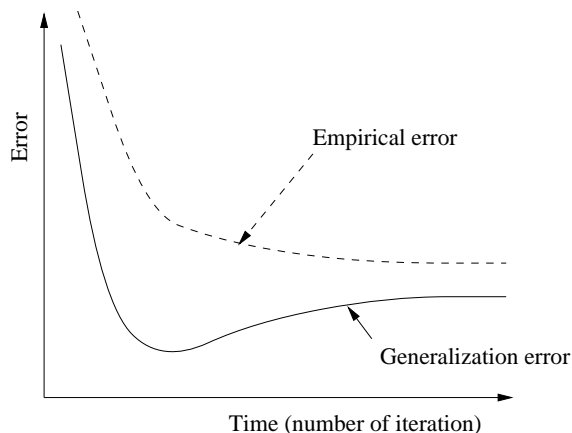


Figure 1: Overtraining in batch learning. The empirical error decreases during iterative training, while the generalization increases after some time point.

for example, a numerical optimization method like the error back-propagation (Rumelhart et al., 1986) is needed to find the optimal parameters. It is very important from the practical and theoretical viewpoints to elucidate the dynamical behavior of a network in training. Especially, the *empirical error*, the objective function of learning, and the *generalization error*, the difference between the target function and its estimate, are of much interest in many researches. For example, the existence of a plateau, which is a long flat interval of a learning curve, has been reported, and a method of avoiding plateaus using the natural gradient has been proposed (Amari, 1998).

In this paper, we focus on *overtraining* (Amari, 1996) as a typical behavior of batch learning in multilayer networks. Overtraining is the increase of the generalization error after some time point in learning (Fig.1). Although the empirical error is a decreasing function in principle, it does not ensure the decrease of the generalization error. Overtraining is not only of theoretical interest but of practical importance, because we can use an early stopping technique if overtraining really exists.

There has been a controversy about the existence of overtraining. Some practitioners assert that overtraining often happens and early stopping methods are proposed (Sarle, 1995). Amari et al. (1996) theoretically analyze overtraining in asymptotic cases and conclude that the effect of overtraining is much smaller than what is believed by practitioners. Their analysis is true if the parameter approaches to the maximum likelihood estimator following the statistical asymptotic theory (Cramér, 1946). However, the usual asymptotic theory cannot be applied in *overrealizable* cases, where the model has surplus hidden units to realize the target function (Fukumizu, 1996; Fukumizu, 1997). In practical applications, since one tends to use a large network to ensure realizability of the target, it is very possible that the model has almost surplus hidden units. The

existence of overtraining has still been an open problem in such cases.

There have been other studies on overtraining. Baldi and Chauvin (1991) theoretically analyze the generalization error of two-layer linear networks estimating the identity function from noisy data. Although they show overtraining phenomena under some conditions, their analysis is very limited in that the overtraining can be observed only when the variance of the noise in the training sample is different from that of the validation sample. In researches from the viewpoint of statistical mechanics, overtraining or overfitting is also reported in the learning of neural networks (Krough & Hertz, 1992). However, the effect of overfitting in their analysis is very small if the number of the data is sufficiently larger than the number of the parameters. There can be other reasons for strong overtraining observed in practical applications.

The main purpose of this paper is to discuss overtraining of multilayer networks in connection with overrealizability. We investigate two models: one is multilayer perceptrons with the sigmoidal activation functions, and the other is linear neural networks, which we introduce for the simplicity of analysis. Through experimental results on these two models, we can conjecture that overtraining occurs in overrealizable scenarios. We mathematically verify this conjecture in the case of linear neural networks under some conditions.

This paper is organized as follows. Section II gives basic definitions and terminology. Section III shows experimental results concerning overtraining of multilayer perceptrons and linear neural networks. In Section IV, we theoretically show the existence of overtraining in overrealizable cases of linear neural networks. Section V includes concluding remarks.

2 Preliminaries

2.1 Framework of statistical learning

In this section, we present basic definitions and terminologies used in this paper. A feed-forward neural network model can be described as a parametric family of functions $\{\mathbf{f}(\mathbf{x}; \boldsymbol{\theta})\}$ from \mathbb{R}^L to \mathbb{R}^M , where \mathbf{x} is an input vector and $\boldsymbol{\theta}$ is a parameter vector. A three-layer network with H hidden units is defined by

$$f^i(\mathbf{x}; \boldsymbol{\theta}) = \sum_{j=1}^H w_{ij} \varphi \left(\sum_{k=1}^L u_{jk} x_k + \zeta_j \right) + \eta_i, \quad (1 \leq i \leq M) \quad (1)$$

where $\boldsymbol{\theta} = (w_{11}, \dots, w_{MH}, \eta_1, \dots, \eta_M, u_{11}, \dots, u_{HL}, \zeta_1, \dots, \zeta_H)$. The function $\varphi(t)$ is called an activation function. In the case of a multilayer perceptron (MLP), the sigmoidal function

$$\varphi(t) = \frac{1}{1 + e^{-t}} \quad (2)$$

is used for the activation function.

We use such models for regression problems, assuming that an output of the target system is observed with a measurement noise. A sample (\mathbf{x}, \mathbf{y}) from the target system satisfies

$$\mathbf{y} = \mathbf{f}(\mathbf{x}) + \mathbf{v}, \quad (3)$$

where $\mathbf{f}(\mathbf{x})$ is the *target function*, which is unknown to a learner, and \mathbf{v} is a random vector subject to $N(0, \sigma^2 I_M)$, where $N(\mu, \Sigma)$ is a normal distribution with μ as its mean and Σ as its variance-covariance matrix. We write I_M for the $M \times M$ unit matrix. An input vector \mathbf{x} is generated randomly with its probability density function $q(\mathbf{x})$. Training data $\{(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}) | \nu = 1, \dots, N\}$ are independent samples from the joint distribution of $q(\mathbf{x})$ and eq.(3). The parameter $\boldsymbol{\theta}$ is estimated so that a neural network can give a good estimate of the target function $\mathbf{f}(\mathbf{x})$.

We discuss the least square error (LSE) estimator. The purpose of training is to find the parameter that minimizes the following *empirical error*:

$$E_{emp} = \sum_{\nu=1}^N \|\mathbf{y}^{(\nu)} - \mathbf{f}(\mathbf{x}^{(\nu)}; \boldsymbol{\theta})\|^2. \quad (4)$$

Generally, the LSE estimator cannot be calculated analytically if the model is nonlinear. Some numerical optimization method is needed to obtain an approximation. A widely-used one is the steepest descent method, which iteratively updates the parameter vector $\boldsymbol{\theta}$ in the steepest descent direction of the error surface defined by $E_{emp}(\boldsymbol{\theta})$. The learning rule is given by

$$\boldsymbol{\theta}(t+1) = \boldsymbol{\theta}(t) - \beta \frac{\partial E_{emp}}{\partial \boldsymbol{\theta}}, \quad (5)$$

where β is a learning constant. In this paper, we discuss only batch learning, in which we calculate the gradient using all the training data. There are many researches on on-line learning, in which the parameter is always updated for a newly generated data (Heskes & Kappen, 1991)

The performance of a network is often evaluated by the generalization error, which represents the average error between the target function and its estimate. More precisely, it is defined by

$$E_{gen} \equiv \int \|\mathbf{f}(\mathbf{x}; \boldsymbol{\theta}) - \mathbf{f}(\mathbf{x})\|^2 q(\mathbf{x}) d\mathbf{x}. \quad (6)$$

It is easy to see

$$\int \int \|\mathbf{y} - \mathbf{f}(\mathbf{x}; \boldsymbol{\theta})\|^2 p(\mathbf{y}|\mathbf{x}) q(\mathbf{x}) d\mathbf{y} d\mathbf{x} = M\sigma^2 + E_{gen}. \quad (7)$$

The empirical error divided by N approaches to the left hand side of eq.(7) if N goes to infinity. The minimization of the empirical error roughly approximates the minimization of the generalization error. However, they are not exactly the same, and the decrease of the empirical error during learning does not ensure the decrease of the generalization error. It is extremely important to elucidate the dynamical behavior of the generalization error. A curve showing the empirical/generalization error as a function of time is called a learning curve.

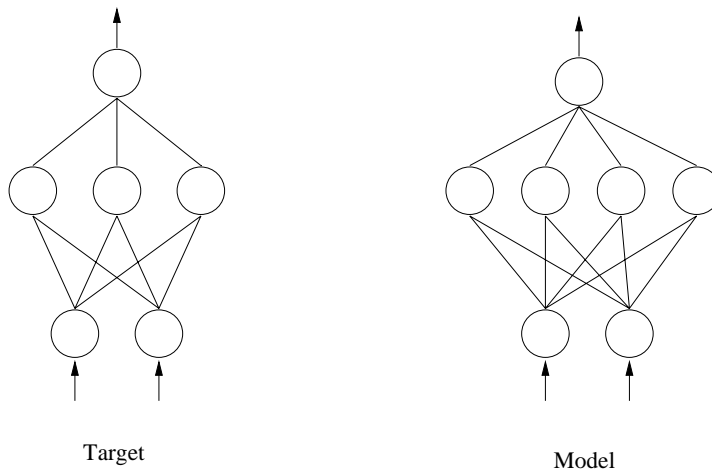


Figure 2: Illustration of an overrealizable case.

2.2 Overrealizable learning scenario

In our theoretical discussion, we consider the case where $\mathbf{f}(\mathbf{x})$ is perfectly realized by the prepared model; that is, there is a true parameter $\boldsymbol{\theta}_0$ such that $\mathbf{f}(\mathbf{x}; \boldsymbol{\theta}_0) = \mathbf{f}(\mathbf{x})$. If the target function $\mathbf{f}(\mathbf{x})$ is realized by a network with a smaller number of hidden units, we call it *overrealizable* (see Fig.2), following the terminology of Saad and Solla (1995). Otherwise, we call it *regular*. We focus on the difference of behaviors between these two cases.

One of the special properties of overrealizable cases is that the true parameter is not unique. The set of parameters that realize the target function is not a point but a union of high-dimensional manifolds. The usual asymptotic theory cannot be applied to such cases. In the presence of measurement noise in the output data, the LSE estimator is located a little apart from the high-dimensional set.

The overrealizable assumption is generally too strong in real problems. However, we usually use a model with a sufficient number of hidden units to avoid a large bias of the model. Therefore, it is highly probable that the model has almost surplus hidden units to realize the target. The question is which assumption describes better in the presence of almost surplus hidden units. We consider this problem in Section 3 through experiments.

We note again that this paper discusses asymptotic situations. While we consider overrealizable cases, the number of data is still sufficiently larger than the number of parameter. This represents redundancy different from the case where the number of data is larger than the number of parameters, which is often discussed in researches with statistical mechanics (e.g. Krough & Hertz, 1992).

2.3 Linear neural networks

We introduce three-layer linear neural networks as a model whose theoretical analysis is possible. We must be very careful in discussing experimental results on learning of multilayer perceptrons especially in overrealizable cases, in which there exists an almost flat subvariety around the global minima (Fukumizu, 1997). The convergence of learning is much slower than the convergence in regular cases because the gradient is very small around this subvariety. In addition, of course, learning with a gradient method suffers from local minima, which is a common problem to all nonlinear models. We cannot exclude their effects, and this often makes derived conclusions obscure. On the other hand, the theoretical analysis of linear neural networks is possible (Baldi & Hornik, 1995; Fukumizu, 1997), and the dynamics of learning will be analyzed in Section 4.

A linear neural network has the identity function for $\varphi(t)$. In this paper, we do not use bias terms in linear neural networks for simplicity. Thus, a linear neural network is defined by

$$\mathbf{f}(\mathbf{x}; A, B) = B A \mathbf{x}, \quad (8)$$

where A is a $H \times L$ matrix and B is a $M \times H$ matrix. We assume $H \leq L$ and $H \leq M$ throughout this paper. Although $\mathbf{f}(\mathbf{x}; A, B)$ is a linear function, from the assumption on H the model is not equal to the set of all linear maps from \mathbb{R}^L to \mathbb{R}^M , but is the set of linear maps of rank not greater than H . Then, the model is not equivalent to the usual linear model $\mathbf{f}(\mathbf{x}; C) = C \mathbf{x}$ ($C : M \times L$ matrix). In this sense, the three-layer linear neural network model is the simplest multilayer model.

The parameterization in eq.(8) has trivial redundancy. If we multiply a nonsingular $H \times H$ matrix from the right of B and its inverse from the left of A , the resultant map is the same. Given a linear map of rank H , the set of parameters that realize the map consists of a $H \times H$ manifold. However, we can eliminate this redundancy if we restrict the parameterization as the first H rows of A make the unit matrix. More essential redundancy in the parameterization arises when the rank of the map is less than H . Even if we use the above restriction, the set of parameters that realize such a map of smaller rank is still high-dimensional. Then, in the case of linear neural networks, we say overrealizable if the rank of the target function is less than H .

3 Learning curves – experimental study –

In this section, we experimentally investigate the generalization error of multilayer perceptrons and linear neural networks to see their actual behaviors, emphasizing on overtraining.

The steepest descent method in multilayer perceptron leads the well-known error back propagation rule (Rumelhart, 1986). To avoid the problems discussed in Section 2.3 as much as possible, we adopt the following configuration in our

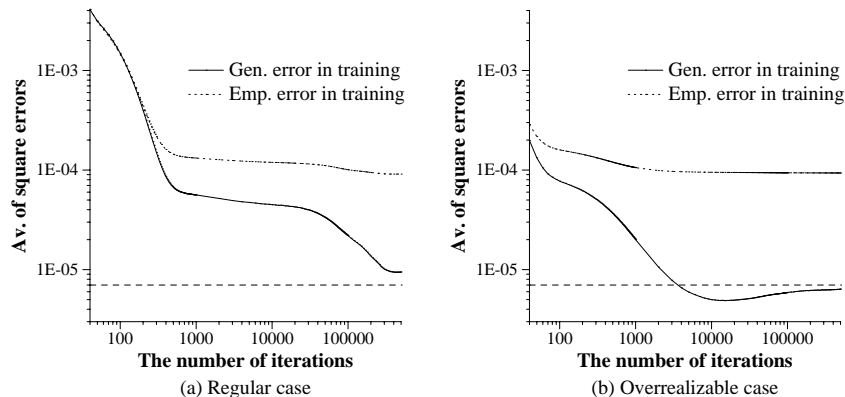


Figure 3: Average learning curves of MLP. The dashed line shows the theoretical expectation of the generalization error of the LSE estimator in the case that the target function is regular and the usual asymptotic theory is applicable.

experiments. The model in use has 1 input unit, 2 hidden units, and 1 output unit. A set of training data has 100 examples. For a fixed set of training data, 30 different values are tried for an initial parameter vector. We select the trial that gives the least empirical error after 500000 iterations. The output data include the observation noise subject to $N(0, 10^{-4})$. The constant zero function is used as an overrealizable target function, and $f(x) = s(x + 1) + s(x - 1)$ is used as a regular target. Figure 3 shows the average of generalization errors over 30 different simulations changing the set of training data. It shows clear overtraining in the overrealizable case. On the other hand, the learning curve in the regular case shows no meaningful overtraining.

It is known that the LSE estimator of the linear neural network model is analytically solvable (Baldi & Hornik, 1995). Although it is practically absurd to use the steepest descent method, our interest is not the LSE estimator itself but the dynamics of learning. Therefore, we investigate the steepest descent learning of linear neural networks here.

For the training data of a linear neural network, we use the notations;

$$X = \begin{pmatrix} \mathbf{x}^{(1)T} \\ \vdots \\ \mathbf{x}^{(N)T} \end{pmatrix}, \quad Y = \begin{pmatrix} \mathbf{y}^{(1)T} \\ \vdots \\ \mathbf{y}^{(N)T} \end{pmatrix}, \quad (9)$$

where T denotes transposition. Then, the empirical error can be written as

$$E_{emp} = \text{Tr}[(Y - XA^T B^T)^T (Y - XA^T B^T)], \quad (10)$$

The batch learning rule of a linear neural network is given by

$$\begin{cases} A(t+1) &= A(t) + \beta\{B^T Y^T X - B^T B A X^T X\}, \\ B(t+1) &= B(t) + \beta\{Y^T X A^T - B A X^T X A^T\}. \end{cases} \quad (11)$$

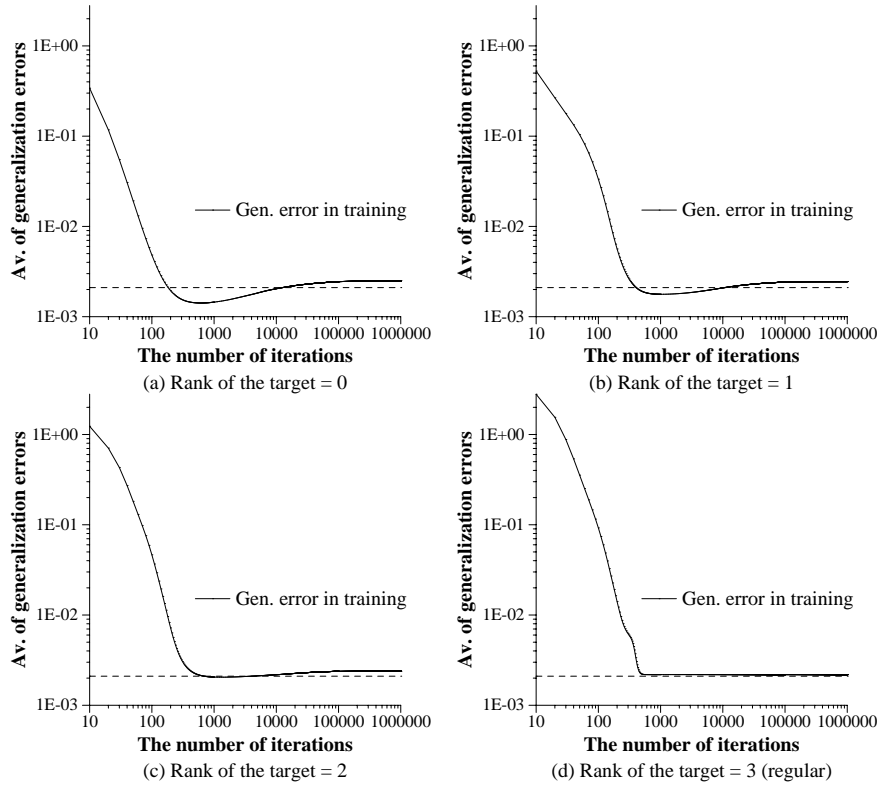
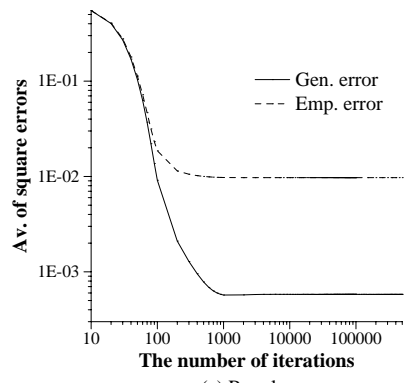


Figure 4: Average learning curves of linear neural networks. The dotted line shows the theoretical expectation of the generalization error of the LSE estimator for the regular target.

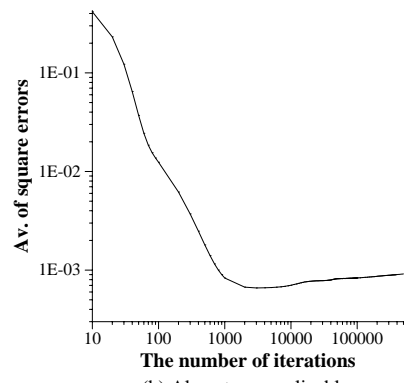
It is also known that all the critical points but the global minimum of E_{emp} are saddle points (Baldi & Hornik, 1995). Therefore, if we theoretically consider a continuous-time learning equation, it converges to the global minimum for almost all initial conditions.

Figure 4 shows the average of the generalization errors for each target of rank 0,1,2, and 3. Each curve is the average over 100 simulations with various training data from the same probability. The number of input units, hidden units, and output units are 5, 3, and 5 respectively. The output samples include the observation noise subject to $N(0, 10^{-2}I_5)$. The number of training data is 100 in each simulation. All the overrealizable cases show clear overtraining, while the regular case does not show meaningful overtraining. We can also see that overtraining is stronger when the rank of the target is smaller.

In the next experiment with the MLP model, we use the error function,



(a) Regular



(b) Almost overrealizable