Exploiting Heavy Tails in Training Times of Multilayer Perceptrons. A Case Study with the UCI Thyroid Disease Database

Manuel Cebrián and Iván Cantador*

Abstract

The random initialization of weights of a multilayer perceptron makes it possible to model its training algorithm as a Las Vegas algorithm, i.e. a randomized algorithm which stops when some required training error is obtained and whose execution time is a random variable. This modeling is used to perform a case study on a well-known pattern recognition benchmark: the UCI Thyroid Disease Database. Empirical evidence is presented of the training time probability distribution exhibiting a heavy tail behavior, meaning a big probability mass of long executions. This fact is exploited to reduce the training cost by applying two simple restart strategies. The first assumes full knowledge of the distribution yielding a 40% cut down in expected time with respect to the training without restarts. The second, assumes null knowledge, yielding a reduction ranging from 9% to 23%.

*The authors are with the Departamento de Ingeniería Informática, Escuela Politécnica Superior, Universidad Autónoma de Madrid, 28049 Madrid, Spain, fax number: (+34) 914 972 235, e-mail: {manuel.cebrian, ivan.cantador}@uam.es.

Keywords: Multilayer Perceptron, Training Time Distribution, Heavy Tail Distribution, Restart Strategy, UCI Thyroid Disease Database.

1 Introduction

The training time of a Multilayer Perceptron (MLP), understood as the time needed to obtain some required training error, is a random variable which depends on the random initialization of the MLP weights.

These weights are commonly initialized according to a given probability distribution, having this choice a significant impact on the training time distribution (see Delashmit & Manry 2002, Duch, et al. 1997, LeCun, et al. 1998). To address this problem, some weight initialization methods have been proposed (e.g. Duch et al. 1997, Weymaere & Martens 1994). They try and reduce the training time by the design of probability distributions based on knowledge about the training set.

In this correspondence, a simpler and more general approach which does not make use of the mentioned information is presented. To do this, we model the learning process of a MLP as a *las Vegas* algorithm (Luby, et al. 1993), i.e. a randomized algorithm which mets three conditions: (i) it stops when some predefined training error δ is obtained, (ii.) the only measurable observation is the training time, and (iii.) it is only possible to have either full or null knowledge about the training time probability distribution.

Using this modeling, we perform a case study with the UCI Thyroid Disease database[1], revealing that the time distribution for learning this pattern recognition benchmark belongs to the *heavy tail* family. This family of distributions is regarded as non-standard for its big probability mass of arbitrary long

[1]The UCI Repository of Machine Learning Databases, available online at http://www.ics.uci.edu/~mlearn/MLRepository.html

executions.

We make use of formal and experimental results which prove that the expected execution time of a random algorithm with such underlying distribution can be reduced by using *restart strategies* (Gomes 2003). This work adapts these strategies to the MLP context: the MLP is trained during a number of epochs t_1. If the required training error δ is achieved before t_1, then the execution finishes. Otherwise, we initialize again the weights in a randomized way, and re-train the MLP during t_2 epochs. The process is iteratively repeated until the training error δ is reached.

Two different strategies are applied for the determination of optimal restarting times. The first assumes full knowledge of the distribution yielding a 40% cut down in expected time with respect to the training without restarts. The second, assumes null knowledge, yielding a reduction ranging from 9% to 23%. As far as we know this is the first research towards appliying restart strategies to MLPs in a principled way.

The rest of the paper is organized as follows. Section 2 presents the Thyroid Disease database and provides evidence of heavy tail behavior when a MLP is trained on it. Section 3 tests the condition to be satisfied by the probability distribution to profit from restart strategies, providing an empirical evaluation of two strategies on a particular case study. Finally, some conclusions and future research lines are given in section 4.

2 A case study: the UCI Thyroid Disease Database

To motivate the use of restarts in MLP learning, we firstly present the existence of a high variability in its training time, indicative of an underlying heavy tail behavior. The evaluation was performed using the UCI Thyroid Disease database, as a case study.

Table 1 shows the expectations, deviations (and its ratio) of the numbers of epochs T spent in building a MLP of a hidden layer with $n = 1, \dots, 8$ units. The training used a gradient descent algorithm with a required training error $\delta = 0.02$. The results were calculated by training the algorithm $1,000$ times for each n, using 10-fold 10-cross validation.

n	1	2	3	4	5	6	7	8
$E[T]$	8551.7	5516.8	888.5	2339.7	1680.2	587.6	482.4	490.5
$\sigma[T]$	2547.5	3885.6	1565.5	2848.8	1355.6	55.1	296.9	464.1
$\sigma[T]/E[T]$	30%	70%	156%	106%	79%	10%	60%	95%

Table 1: Expectation, deviation (and its ratio) of the number of epochs T spent in the building of a MLP with n hidden units and training error $\delta = 0.02$.

The obtained deviations are very large in relation to the expectations for the majority of the architectures. For the rest of the experiments, we shall use a MLP with $n = 3$ hidden units, which has the highest relative variability. This will serve as a proof of concept, altough the same behavior is observed in MLPs with other number of hidden units.

Following, we give visual evidence that T is heavy tailed, i.e. that the probability of the training time T being greather than some number of epochs t has polynomial decay, *viz.* $P[T > t] \sim C.t^{-\alpha}$, where $\alpha \in (0, 2)$, C is some constant, and $t > 0$.

Figure 1 presents a log-log plot of $P[T > t]$ for the 10% largest values ($t > 3,000$). The plot confirms the polynomial decay by displaying a straight line with slope $-\alpha$. This is because, for sufficiently large t, $\log P[T > t] = -\alpha \log C.t \Rightarrow \log P[T > t]/\log C.t \approx -\alpha$.

Finally, we verify that α belong to the $(0, 2)$ interval. For this, we compute the Hill's (1975) estimator:

$$\hat{\alpha}_r = \left(r^{-1} \sum_{j=1}^{r} \ln T_{m,m-j+1} - \ln T_{m,m-r} \right),$$

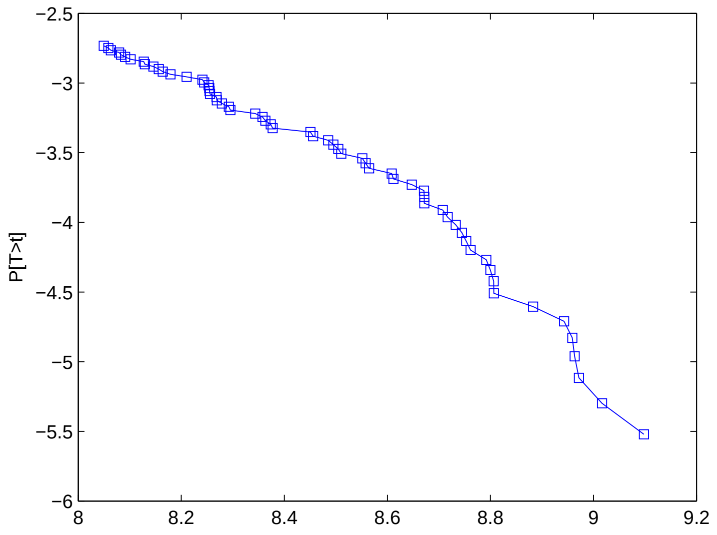

Figure 1: A log-log plot of $P[T > t]$ as a function of t (in epochs).

where $T_{m,1} \leq T_{m,2} \leq \ldots \leq T_{m,m}$ are the m ordered training completion times, and $r < m$ is a cutoff that allows to observe only the highest values (the tail). We use the typical cutoff $r = 0.1m$ and obtain $\hat{\alpha}_r = 1.942$, consistent with our hypothesis.

This polynomial decay, whichyields a big probability mass for long executions, is due to the fact that certain initial weights entail a convergence to local minima of the target function, requiring very long (even infinite) training periods, while others yield a convergence to global minima in a few epochs.

3 Restart strategies

A las Vegas algorithm may profit from restarting if, at some moment of the execution τ, the expected completion time conditioned to the already employed execution time ($E[T - \tau | T > \tau]$) is larger than the (unconditioned) expected

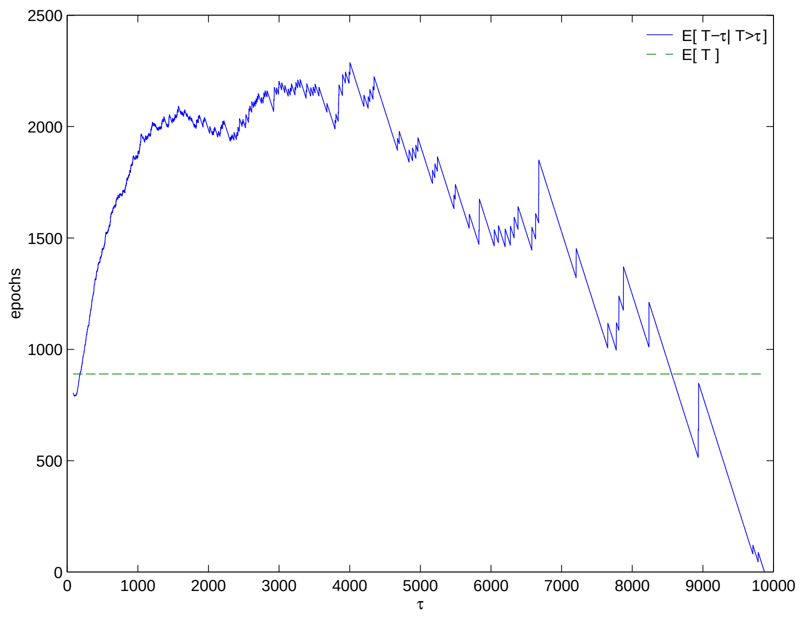

Figure 2: $E[T - \tau | T > \tau]$ as a function of τ, $E[T]$ serves as the baseline.

completion time ($E[T]$), i.e. if $\exists \tau, \quad E[T] < E[T - \tau | T > \tau]$ (see van Moorsel & Wolter 2004).

Figure 2 displays that the majority of τ values which met the condition for the PMC to profit of restart strategies.

3.1 Restart strategies when the distribution is known

Luby et al. (1993) proves the existence of an optimal restart strategy for a las Vegas algorithm which minimizes the expected running time when the execution time distribution $q(t) = \Pr(T < t)$ is assumed known.

This optimal strategy is of the form $t_i = t^* \ \forall i$, where

$$t^* = \arg\min_t E[S_t] = \arg\min_t \frac{1}{q(t)} \left(t - \sum_{t' < t} q(t') \right) \qquad (1)$$

and S_t is the restart strategy where $t_i = t \ \forall i$ for some t.

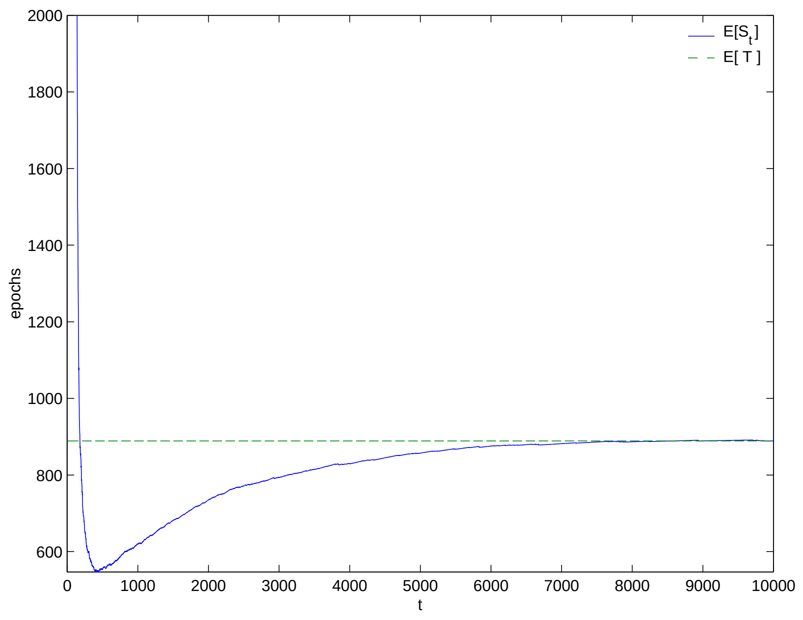

Figure 3: Expected training time using the strategy S_t with $t \in [100,\ 10,000]$, $E[T]$ servers as the baseline.

Simple calculations yield $t^* = 418$, with an optimal expected time $E[S_t^*] = 546.876$. This provides a 40% cut down in expected time with respect to the training without restarts. Figure 3 displays the expected time for strategies of the form S_t with $t \in [100,\ 10,000]$. As it can be seen, many non-optimal t choices provide a time reduction as well.

3.2 Restart strategy when the time distribution is unknown

In some scenarios it is not possible to assume full knowledge of the distribution, e.g. if the MLP is to be trained a single time. In this subsection we assume null knowledge.

Again Luby et al. (1993) prove the existence of an optimal strategy for this assumption, and Walsh (1999) derives a simpler variant of the former which is

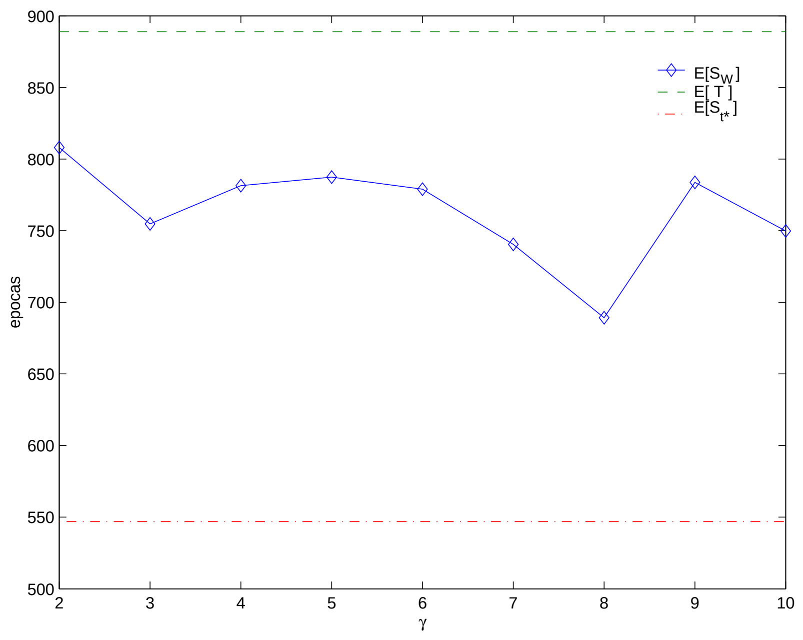

Figure 4: Expected training time using the Walsh strategy $E[S_W]$ for $\gamma = 1, 2, \ldots, 10$, $E[S_t^*]$ and $E[T]$ serve as baselines.

commonly used in practical applications. The Walsh strategy S_W is defined as $t_i = \gamma^{i-1}$, $\gamma > 1$. This strategy benefits of a high probability of success when $t_i = \gamma^{i-1}$ is near to t^*. Increasing t_i geometrically makes it sure to reach t^* in a few generations, expecting to reach error δ within few restarts after the value of t_i surpasses the optimal.

Figure 4 displays the expected values of S_W using several standard γ values $\gamma = 2, 3, \ldots, 10$. Training is speeded with all choices, with improvements ranging from 9% ($\gamma = 2$) to 23% ($\gamma = 8$). The expected times were computed running $1,000$ times the training algorithm for each γ.

4 Conclusions and future work

In this work, MLP training algorithm is modeled as a las Vegas algorithm, performing a case study on the UCI Thyroid Disease Database. We give statistical evicence of that the probability distribution of the training time belongs to the heavy tail family, meaning an polynomial probability decay for long executions. This property is exploited to reduce the training cost by two simple strategies. The first assumes full knowledge of the distribution yielding a 40% cut down in expected time with respect to the training without restarts. The second, assumes null knowledge, yielding a reduction ranging from 9% to 23%.

As a future research, we plan to determine whether further improvements can be obtained by relaxing the two las Vegas algorithms assumptions (see sect. 1). This could make it possible to incorporate dynamic restart strategies (see Kautz, et al. 2002) capable of exploiting epoch-by-epoch information about the training time distribution, using various algorithm behavior measurements besides the execution time.

References

W. Delashmit & M. Manry (2002). 'Enhanced robustness of multilayer perceptron training'. In *Proceedings of the Thirty-Sixth Asilomar Conference on Signals, Systems and Computers*, pp. 1029–1033.

W. Duch, et al. (1997). 'Initialization and Optimization of Multilayered Perceptrons'. In *3rd Conference on Neural Networks and Their Applications*, pp. 99–104, Kule, Poland.

C. Gomes (2003). *Constraint and Integer Programming: Toward a Unified Methodology*, chap. Complete randomized backtrack search, pp. 233–283. Kluwer Academics.

B. Hill (1975). ‘A Simple General Approach to Inference About the Tail of a Distribution’. *The Annals of Statistics* **3**(5):1163–1174.

H. Kautz, et al. (2002). ‘Dynamic restart policies’. *Proceedings AAAI-2002* pp. 674–681.

Y. LeCun, et al. (1998). ‘Efficient BackProp’. *Lecture Notes in Computer Science* **1524**:5–50.

M. Luby, et al. (1993). ‘Optimal speedup of Las Vegas algorithms’. *Proceedings of the 2nd Israel Symposium on the Theory and Computing Systems* pp. 128–133.

A. van Moorsel & K. Wolter (2004). ‘Analysis and Algorithms for Restart’. *Proceedings of the 1st International Conference on Quantitative Evaluation of Systems* pp. 195–204.

T. Walsh (1999). ‘Search in a Small World’. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pp. 1172–1177.

N. Weymaere & J. P. Martens (1994). ‘On the Initialization and Optimization of Multilayer Perceptrons’. *IEEE Transactions on Neural Networks* **5**:738–751.