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 process 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 time 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%.

Keywords: Stochastic Modeling, Multilayer Perceptron, 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 attempt to reduce
the training time by applying different probability distributions on the initial weights of the MLP 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 meets three conditions: (i) it stops when some pre-defined training error $\delta$ is obtained, (ii) its only measurable observation is the training time, and (iii) it only has 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\footnote{The UCI Repository of Machine Learning Databases, available online at http://www.ics.uci.edu/\textasciitilde mlearn/MLRepository.html}, revealing that the time distribution for learning this pattern recognition benchmark belongs to the heavy tail distribution family. This type of distributions is regarded as non-standard for its big probability mass of arbitrary long values.

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 $\delta$ 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 $\delta$ is reached, being $t_i$ the restart threshold (in epochs) after $i − 1$ restarts have been performed.

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%.

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 the 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 single hidden layer MLP with $n = 1, \ldots, 8$ units. The MLP was trained using the well-known Back-Propagation technique with a
target training error $\delta = 0.02$. The results shown were computed using 10-fold cross validation.

The obtained deviations are very large respect to the expectations for most 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, although the same behavior is observed in MLPs with other number of hidden units.

In the following, we give visual evidence that $T$ is heavy tailed, i.e. that the probability of the training time $T$ being greater 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 $\alpha$ belong to the $(0, 2)$ interval by computing the Hill’s (1975) estimator:

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

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 $\hat{\alpha}_r = 1.942$, which is consistent with our hypothesis.

This polynomial decay, which yields 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 point of the execution $\tau$, the expected completion time conditioned to the already employed execution time ($E[T - \tau | T > \tau]$) is larger than the (unconditioned) expected

| $n$ | $E[T]$ | $\sigma[T]$ | $\sigma[T]/E[T]$ |
|-----|--------|-------------|-----------------|
| 1   | 8551.7 | 2547.5      | 30%             |
| 2   | 5516.8 | 3885.6      | 70%             |
| 3   | 888.5  | 1565.5      | 156%            |
| 4   | 2339.7 | 2848.8      | 106%            |
| 5   | 1680.2 | 1355.6      | 79%             |
| 6   | 587.6  | 55.1        | 10%             |
| 7   | 482.4  | 296.9       | 60%             |
| 8   | 490.5  | 464.1       | 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 training algorithm was run 1,000 times for each number of hidden units.
completion time \(E[T]\), i.e. if \(\exists \tau, E[T] < E[T - \tau | T > \tau]\) (see van Moorsel & Wolter 2004).

Figure 2 shows that the majority of \(\tau\) values met the condition for the MLP to profit of restart strategies.

3.1 Restart strategies when the distribution is known

Luby et al. (1993) prove 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 a fixed restart threshold for all iteration of the form \(t_i = t^* \forall i\), where

\[ t^* = \text{arg min}_t E[S_t] = \text{arg min}_t \frac{1}{q(t)} \left( t - \sum_{t' < t} q(t') \right) \tag{1} \]

and \(S_t\) is the restart strategy where \(t_i = t \forall i\) for some \(t\). We assume some discretization of the time, so that expressions like \(t' < t\) make sense.

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 (see Table 1). 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 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 $t^\ast$. Increasing $t_i$ geometrically makes it sure to reach $t^\ast$ in a few generations, expecting to reach error $\delta$ within few restarts after the value of $t_i$ surpasses the optimal.

Figure 4 displays the expected values of $S_W$ using several standard $\gamma$ 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 $\gamma$. 

![Figure 2: $E[T - \tau] | T > \tau$ as a function of $\tau$, $E[T]$ serves as the baseline.](image-url)
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 visual and numerical evidence that the probability distribution of the training time belongs to the heavy tail family, meaning a polynomial probability decay for long executions. This property is exploited to reduce the training time 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 las Vegas algorithms assumptions (ii) and (iii) (see section 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.
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.

5 Acknowledgements

This work was partially supported by grant TSI2005-08255-C07-06 of the Spanish Ministry of Education and Science. We would also like to thank Ignacio García for his useful suggestions and comments on this manuscript.

References

W. Delashmit & M. Manry (2002). ‘Enhanced robustness of multilayer perceptron training’. In Proceedings of the 36th Asilomar Conference on Signals, Systems and Computers, pp. 1029–1033.

W. Duch, et al. (1997). ‘Initialization and Optimization of Multilayered Perceptrons’. In Proceedings of the 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’. In *Proceedings of the 18th American Association on Artificial Intelligence*, 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’. In *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’. In *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 16th 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.