Baltzer Journals March 16, 1996

Capacity of the Upstart Algorithm

Ansgar H. L. West[1,2] and David Saad[2]

[1] *Department of Physics, University of Edinburgh, Mayfield Road, Edinburgh EH9 3JZ, United Kingdom*
E-mail: `A.H.L.West@ed.ac.uk`

[2] *Neural Computing Research Group, Aston University, Aston Triangle, Birmingham B4 7ET, United Kingdom*
E-mail: `D.Saad@aston.ac.uk`

The storage capacity of multilayer networks with overlapping receptive fields is investigated for a constructive algorithm within a one-step replica symmetry breaking (RSB) treatment. We find that the storage capacity increases logarithmically with the number of hidden units K without saturating the Mitchison-Durbin bound. The slope of the logarithmic increase decays exponentially with the stability with which the patterns have been stored.

1 Introduction

Since the ground breaking work of Gardner [1] on the storage capacity of the perceptron, the replica technique of statistical mechanics has been successfully used to investigate many aspects of the performance of simple neural network models. However, progress for multilayer feedforward networks has been hampered by the inherent difficulties of the replica calculation. This is especially true for capacity calculations, where replica symmetric (RS) treatments [2] violate the upper Mitchison-Durbin bound [3] derived by information theory. Other efforts [4] break the symmetry of the hidden units explicitly prior to the actual calculation, but the resulting equations are approximations and difficult to solve for large networks. This paper avoids these problems by addressing the capacity of a class of networks with variable architecture produced by a constructive algorithm. In this case, results derived for simple binary perceptrons above their saturation limit [5] can be applied iteratively to yield the storage capacity of two-layer networks.

Constructive algorithms (e.g., [6, 7]) are based on the idea that in general it is *a priori* unknown how large a network must be to perform a certain classification task. It seems appealing therefore to start off with a simple network, e.g., a binary perceptron, and to increase its complexity only when needed. This procedure has the added advantage that the training time of the whole network is relatively short, since each training step consists of training the newly added hidden units only, whereas previously constructed weights are kept fixed. Although constructive algorithm seem therefore rather appealing, their properties are not well understood. The aim of this paper is to analyse the performance of one constructive algorithm, the upstart algorithm [7], in learning random dichotomies, usually referred to as the capacity problem.

The basic idea of the upstart algorithm is to start with a binary perceptron unit with possible outputs {1,0}. Further units are created only if the initial perceptron makes any errors on the training set. One unit may have to be created to correct **WRONGLY ON** errors (where the target was 0 but the actual output is 1) another to correct **WRONGLY OFF** errors (where the target was 1 but the output is 0). If these units still cause errors in the output of the network, more units are created in the next generation of the algorithm until all outputs are correct. Different versions of the upstart algorithm differ in the way new units are connected to the old units and to the output unit. The original upstart algorithm produces a hierarchical network where the number of hidden units tends to increase exponentionally with each generation. Other versions of the upstart algorithm[7] build a two-layer architecture and show only a linear increase of the number of units with each generation, which is in general easier to implement.

We have therefore analysed a non-hierarchical version of the upstart algorithm. Within a one-step replica symmetry breaking (RSB) treatment [8], networks constructed by the upstart algorithm show a logarithmic increase of the capacity with the number of nodes in agreement with the Mitchison-Durbin bound ($\alpha_c \propto \ln K/\ln 2$), whereas the simpler RS treatment violates this bound. Furthermore, the algorithm does not saturate the Mitchison-Durbin bound for zero stability. We further find that the slope of the logarithmic increase of the capacity against network size decreases exponentionally with the stability.

2 Model description and framework

2.1 Definition of the upstart algorithm

The upstart algorithm first creates a binary perceptron (or unit) $\mathcal{D}_0$ which learns a synaptic weight vector $\boldsymbol{W} \in \mathbb{R}^N$ and a threshold θ which minimize the error on a set of p input-output mappings $\boldsymbol{\xi}^\mu \in \{-1,1\}^N \to \zeta^\mu \in \{0,1\}$ $(\mu = 1,\ldots,p)$ from an N–dimensional binary input space to binary targets. The output of the binary perceptron is determined by

$$\sigma^\mu = \Theta\left(\frac{1}{\sqrt{N}}\boldsymbol{W}\cdot\boldsymbol{\xi}^\mu - \theta\right) = \Theta(h^\mu)$$

where $\boldsymbol{\Theta}(x)$ is the Heavyside stepfunction, which is 1 for $x \geq 0$ and 0 otherwise, and h^μ is the activation of the perceptron. The error is defined as

$$E = \sum_\mu \boldsymbol{\Theta}\left[\kappa - 2(\zeta^\mu - 1)h^\mu\right],$$

where κ is the stability with which we require the patterns to be stored. A suitable algorithm (e.g., [9]) will converge to a set of weights $\boldsymbol{W}$ which minimizes the above error. If the set of examples is not linearly separable with a minimum distance κ of all patterns to the hyperplane, the binary perceptron will not be able to classify all patterns correctly, i.e., $\sigma^\mu \neq \zeta^\mu$ for some μ's and the upstart algorithm has to create further daughter units an a hidden layer to realize the mapping. The upstart algorithm therefore creates a binary $\{0,1\}$ output unit $\mathcal{O}$ with threshold one and the initial perceptron $\mathcal{D}_0$ and all further daughter units to be built by the algorithm will form the hidden layer. The first perceptron is then connected to $\mathcal{O}$ with a $+1$ weight, i.e., $\mathcal{O}$ has initially the same outputs as $\mathcal{D}_0$.

The basic idea of the upstart algorithm is to create further daughter units $\mathcal{D}^+$ and $\mathcal{D}^-$ in the hidden layer to correct WRONGLY OFF and WRONGLY ON errors respectively. Consider, for example, the creation of the new hidden unit $\mathcal{D}^-$, which is connected with a large negative weight to $\mathcal{O}$, whose role is to inhibit $\mathcal{O}$. $\mathcal{D}^-$ should be active (1) for patterns for which $\mathcal{O}$ was WRONGLY ON and inactive (0) for patterns for which $\mathcal{O}$ was CORRECTLY ON. Similarly, $\mathcal{D}^-$ ought to be 0 if $\mathcal{O}$ was WRONGLY OFF, in order to avoid further inhibition of $\mathcal{O}$. However, we do not have to train $\mathcal{D}^-$ on patterns for which $\mathcal{O}$ was CORRECTLY OFF, since an active $\mathcal{D}^-$ would only reinforce $\mathcal{O}$'s already correct response. The resulting training sets and the targets of both daughter units are illustrated in Table 1. More formally we define the algorithm upstart II by the following steps which are applied recursively until the task is learned:

Step 0: Follow the above procedure for the original unit $\mathcal{D}_0$ and the creation of the output unit $\mathcal{O}$. Evaluate the number of WRONGLY OFF and WRONGLY ON errors.

Step 1: If the output unit $\mathcal{O}$ of the upstart network of i generations makes more WRONGLY OFF than WRONGLY ON errors, a new unit $\mathcal{D}^-_{i+1}$ is created

Table 1: The targets of the upstart II algorithm depending on the requested target ζ and the actual output σ of the output unit $\mathcal{O}$. The target "$*$" means that the pattern is not included in the training set of $\mathcal{D}^\pm$.

	$\zeta = 1$	$\zeta = 0$
$\sigma = 1$	CORRECTLY ON $\mathcal{D}^+$ $*$ $\mathcal{D}^-$ 0	WRONGLY ON $\mathcal{D}^+$ 0 $\mathcal{D}^-$ 1
$\sigma = 0$	WRONGLY OFF $\mathcal{D}^+$ 1 $\mathcal{D}^-$ 0	CORRECTLY OFF $\mathcal{D}^+$ 0 $\mathcal{D}^-$ $*$

and trained on the training set and targets given in Table 1. If there are more WRONGLY ON than WRONGLY OFF errors, a new unit $\mathcal{D}^+_{i+1}$ is created with training set and targets also given in Table 1. If both kind of errors occur equally, two units $\mathcal{D}^-_{i+1}$ and $\mathcal{D}^+_{i+1}$ are created with training sets and targets as above.

Step 2: The new units are trained on their training sets and their weights are frozen. The units $\mathcal{D}^+_{i+1}$, $\mathcal{D}^-_{i+1}$ are then connected with positive, negative weights to the output unit respectively. The modulus of the weights are adjusted so that $\mathcal{D}^\pm_{i+1}$ overrules any previous decisions if active. The total number of WRONGLY OFF and WRONGLY ON errors of the upstart network of generation $i+1$ is then reevaluated. If the network still makes errors the algorithm goes back to Step 1.

The algorithm will eventually converge as a daughter unit will always be able to correct at least one of the previously misclassified patterns without upsetting any already correctly classified examples.

2.2 Statistical mechanics framework for calculating the capacity limit

Since the upstart algorithm trains only perceptrons, we can apply knowledge of the capacity limit and of the error rate of perceptrons above saturation derived in a statistical mechanics framework to calculate the capacity limit of the upstart II algorithm for an arbitrary number of generations. Below, we briefly review this statistical mechanics calculation and refer the reader to [5] and to previous work [1] for a more detailed treatment.

In the capacity problem the aim is to find the maximum number p of random input-output mappings of binary N–dimensional input vectors $\boldsymbol{\xi}^\mu$ to targets $\zeta^\mu \in \{0,1\}$, which can be realized by a network on average. We assume that each component of the input vectors $\boldsymbol{\xi}^\mu$ is drawn independently with equal probability from $\{-1,1\}$. The distribution of targets is taken to be pattern independent with a possible bias b: $P(\zeta) = \frac{1}{2}(1+b)\delta(1-\zeta) + \frac{1}{2}(1-b)\delta(\zeta)$. We will here only consider an unbiased output distribution for the intial perceptron. The target distributions for daughter units however will in general be biased.

Each binary perceptron is trained stochastically and we only allow weight vector solutions with the minimal achievable error. The error rate, i.e., the number of errors divided by the total number of examples, is assumed to be self-averaging with respect to the randomness in the training set in the thermodynamic limit $N \to \infty$. In this limit the natural measure for the number of examples p is $\alpha = p/N$. With increasing α the weight space of possible solutions shrinks, leaving a unique solution at the capacity limit of the binary perceptron. Above the capacity limit many different weight space solutions with the same error are possible. In general the solution space will be disconnected as two solutions can possibly missclassify different patterns. As α diverges, the solution space becomes increasingly fragmented.

The replica trick is used to calculate the solution space and the minimal error rate averaged over the randomness of the training set. This involves the replication of the perceptron weight vector, each replica representing a different possible solution to the same storage problem. In order to make significant progress, one has further to assume some kind of structure in the replica space. Below the capacity limit, the connectedness of the solution space is reflected by the correctness of a replica symmetric (RS) ansatz. Above the capacity, the disconnectedness of the solution space breaks the RS to some degree. We have restricted ourselves to a one-step replica symmetry breaking (RSB) calculation, which is expected to be at least sufficient for small error rates. The form of the equations for the error rate resulting from the RS and one-step RSB calculations are quite cumbersome and will be reported elsewhere [5, 10]. For the perceptron, the error rate is a function of the output bias b and the load α only.

3 Results of the upstart algorithm

The capacity of an upstart network with K hidden units can now be calculated. The initial perceptron is trained with an example load of α and an unbiased output distribution $b = 0$. The saddlepoint equations and the **WRONGLY ON** and **WRONGLY OFF** error rates are calculated numerically. These error rates determine the load and bias for the unit(s) to be created in the next generation. Now its (their) error rates and the errors of the output unit can in turn be calculated by solving the saddlepoint equations. This is iterated until K units have been built. If the output unit still makes error, we are above the capacity limit of the upstart net with K hidden units and α has to be decreased. On the other hand, if the output unit makes no errors, α can be increased. The maximal α for which the output unit makes no errors defines the saturation point of the network. The capacity limit, defined here as the maximal number of examples per adjustable weight of the network, then becomes simply $\alpha_c(K) = \alpha/K$.

In Fig. 1a we present the storage capacity as a function of the number of hidden units for both a one-step RSB and a RS treatment at zero stability of the patterns ($\kappa = 0$). Whereas one-step RSB predicts a logarithmic increase $\alpha_c(K) \propto \ln(K)$ for large networks, in agreement with the Mitchison-Durbin bound, the results for the RS-theory violate this upper bound[1], i.e., the RS theory fails to predict the qualitative behaviour correctly.

In Fig. 1a we also show that the storage capacity still increases logarithmically with the number of units K for non-zero stability, but with a smaller slope γ.

[1]The violation occurs for $K \geq 180$ and the largest networks in the RS case were $K = 999$.

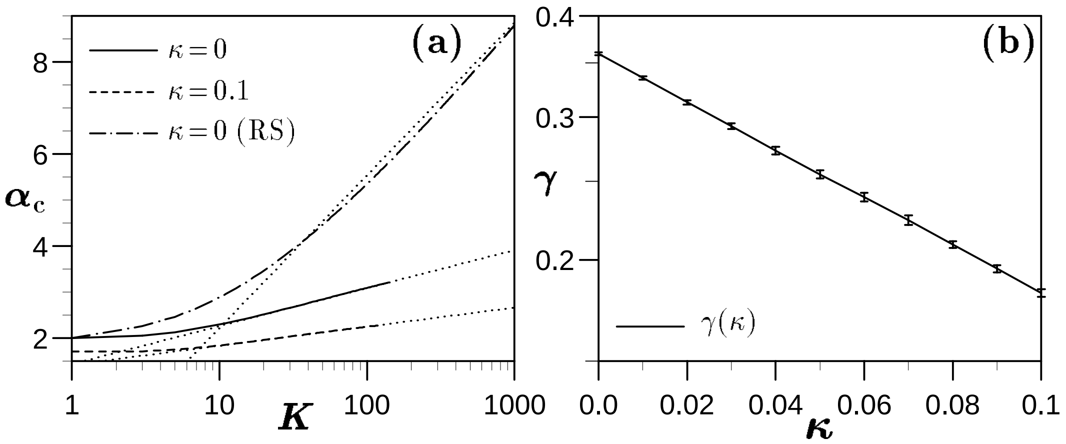

Figure 1: (a) Within the one-step RSB theory, the capacity α_c increases logarithmically with the number of hidden units K for large K for the stabilities $\kappa = 0\,(0.1)$, i.e., $\alpha_c \propto 0.3595\,(0.182) \ln K$ (see superimposed asymptotics). The RS theory violates the Mitchison-Durbin bound (third asymptotic: $\alpha_c \propto \ln K / \ln 2$) for $K \geq 180$. (b) The slope γ of the logarithmic increase of the capacity decreases exponentionally with the stability κ.

Fig. 1b shows the dependence of the slope γ as a function of the stability κ for one-step RSB. The maximal slope for zero stability $\gamma = 0.3595 \pm 0.0015$ does not saturate the Mitchison-Durbin bound $\gamma = 1/\ln 2 \approx 1.4427$, but is about four times lower. With increasing stabilities κ this slope decreases exponentionally $\gamma \propto \exp(-6.77 \pm 0.02\,\kappa)$.

4 Summary and Discussion

The objective of this work has been to calculate the storage capacity of multilayer networks created by the constructive upstart algorithm in a statistical mechanics framework using the replica method. We found that the RS-theory fails to predict the correct results even qualitatively. The one-step RSB theory yields qualitatively and quantitatively correct results over a wide range of network sizes and stabilities.

In the one-step RSB treatment, a logarithmic increase with slope γ of the capacity of the upstart algorithm with the number of units K was found for all stabilities. The slope decreases exponentionally $[\gamma \propto \exp(-6.77\kappa)]$ with the stability κ. It would be interesting to investigate if this result carries over to other constructive algorithms or even to general two-layer networks.

For zero stability the slope of this increase is around four times smaller than the upper bound $(1/\ln 2)$ predicted by information theory. We suggest that this indicates that the upstart algorithm uses its hidden units less effectively than a general two-layer network. We think this is due to the fact that the upstart algorithm uses the hidden units to overrule previous decisions, resulting in an exponential increase of the hidden layer to output unit weights. This is in contrast to general two-layer networks which usually have hidden-output weights of roughly the same order and can therefore explore a larger space of internal representations. For the upstart algorithm a large number of internal representations are equivalent and others cannot be implemented as they are related to erroneous outputs. However, it would be interesting to investigate how other constructive algorithms (e.g., [6]) perform in comparison. A systematic investigation of the storage capacity of constructive algorithms may ultimately lead to a better understanding, and thus possibly to novel, much improved algorithms.

Acknowledgements

AHLW would like to acknowledge gratefully financial support by the EPSRC. This work has been supported by EU grant ERB CHRX-CT92-0063.

References

[1] E. Gardner. J. Phys. A, 21:257–270, 1988.
[2] E. Barkai, D. Hansel and H. Sompolinsky. Phys. Rev. A, 45:4146–4161, 1992.
[3] G. J. Mitchison and R. M. Durbin. Biological Cybernetics, 60:345–356, 1989.
[4] D. Saad. J. Phys. A, 27:2719–2734, 1994.
[5] A. H. L. West, and D. Saad, submitted to J. Phys. A, 1996.
[6] M. Mézard and J.-P. Nadal. J. Phys. A, 22:2191–2203 1989.
[7] M. Frean. Neural Computation, 2:198–209, 1990.
[8] For an overview, see e.g., M. Mézard, G. Parisi and M. G. Virasoro, *Spin Glass Theory and Beyond*, World Scientific, Singapore, 1987.
[9] M. Frean. Neural Computation, 4:946–957, 1992, and references therein.
[10] A. H. L. West and D. Saad. in preparation, 1996.