A Local Optima Network Analysis of the Feedforward Neural Architecture Space

Isak Potgieter  
Department of Computer Science  
University of Pretoria  
Pretoria, South Africa  
isak.potgieter@gmail.com

Christopher W. Cleghorn  
School of Computer Science and Applied Mathematics  
University of the Witwatersrand  
Johannesburg, South Africa  
christopher.cleghorn@wits.ac.za

Anna S. Bosman  
Department of Computer Science  
University of Pretoria  
Pretoria, South Africa  
anna.bosman@up.ac.za

Abstract—This study investigates the use of local optima network (LON) analysis, a derivative of the fitness landscape of candidate solutions, to characterise and visualise the neural architecture space. The search space of feedforward neural network architectures with up to three layers, each with up to 10 neurons, is fully enumerated by evaluating trained model performance on a selection of data sets. Extracted LONs, while heterogeneous across data sets, all exhibit simple global structures, with single global funnels in all cases but one. These results yield early indication that LONs may provide a viable paradigm by which to analyse and optimise neural architectures.

Index Terms—neural architectures, fitness landscapes, local optima networks

I. INTRODUCTION

Much remains unclear about the relationship between neural network architecture, i.e. the number and arrangement of network nodes, and trained model performance. In practise, much of neural architecture optimisation is done empirically. Deeper and, particularly, wider neural networks have been found to be resilient to overfitting [1], [2], improving model capacity to approximate continuous functions [3]–[5] and to consistently converge [6], [7]. Recent findings have shown that larger networks rely on favourably seeded subnetworks [7], [8]. Lacking a priori knowledge about what makes performant subnetworks special, however, such findings may be interpreted as a prescription to train overparameterised networks [7], containing far more parameters than inherently required for optimal training. The gain in model performance comes at a significant computational cost, making deep neural networks time and energy inefficient. This provides motivation to seek new methods with which to characterise the neural architecture space.

Solutions in a search space and the relationships between them can be characterised by introducing a topology using the notion of a fitness landscape [9], with appropriate definitions of a solution’s fitness and its neighbourhood. Ochoa et al. [10] introduced a derivative of the fitness landscape, a local optima network (LON), narrowing the focus to local optima, which comprise the LON nodes. Nodes are connected in a directed graph by edges weighted based on the probability of a perturbation from the starting node resulting in an escape to the basin of the destination node [10], [11]. LONs have been used to study NK-landscapes [10], difficulty phase transitions in combinatorial optimisation [12], and to characterise feature selection algorithms [11]. The use of a LON may enhance analysis of the global landscape structure by aiding visual inspection and yielding a range of numerical landscape features. Combined feature assessment of landscape features enhances search space characterisation, search hardness estimation and optimisation.

As an initial proof of concept of the application of LON analysis to the neural architecture space, the search space considered in this study comprises all possible feedforward neural network architectures with a depth of up to 3 hidden layers, each with up to 10 neurons, trained on a given data set. The fitness of all solutions is measured by taking the mean performance of the corresponding trained model on test data over 30 runs. The evaluation is repeated over 9 different data sets comprising classification and regression tasks, and the results are compared. From each resulting fitness landscape, a LON is derived and its structure investigated to establish the viability of the use of LONs for neural architecture search and optimisation. The following novel results are established:

- A high local optimum fitness standard deviation is related to low modality, which improves the relative ease of fitness optimisation. Low standard deviation at low fitness levels, related to high modality, signals that a model is not sufficiently powerful for a given task.
- Node and edge counts are linearly correlated beyond low extremes. Higher modality involves a higher number of smaller basins, so that a higher proportion of basins are out of reach of any particular candidate solution.
- Edges are more likely to be fitness improving than deteriorating, coinciding with a general increase in local optima basin size.
- LONs differ notably between data sets in terms of node and edge count, fitness distribution, basin size distribution and global optimum incoming strength. The solution space is thus heterogeneous and dependent on the particular data set.
- Despite the diverging LON characteristics, all data sets except one produce a single global sink. The simple funnel structure suggests that simple iterative local optimisers may effectively be used to find near optimal solutions.

This study maintains a focus on densely connected
feedforward neural networks and does not consider alternative architectural arrangements, such as recurrent or partial connections. The search space is restricted to the neural architecture, excluding the weight space and selection of activation function, while acknowledging that both have been shown to have great bearing on model performance. To control for variance in performance due to randomised weight initialisation, the mean fitness over 30 models trained per architecture is used. The range of architectures evaluated and the data sets used are relatively small in size, due to computational constraints. These selections serve to establish initial viability of local optima network analysis in the present domain by fully enumerating the space over multiple data sets.

The rest of the paper is organised as follows. Section II discusses fitness landscapes. Section III provides the background on LONs. Section IV details the neural network configuration used in the study. Section V discusses the datasets used. Section VI presents the empirical results of the study. Finally, Section VII concludes the paper.

II. FITNESS LANDSCAPES

A fitness landscape is formally defined by three components, namely a set of solutions $S$, a notion of a neighbourhood $N(s)$ of each solution $s \in S$, and a fitness function $f(s)$ that assigns a fitness value to each solution [9]. The approach of assigning fitness values to candidate solutions has been adapted for optimisation problems. To analyse the interrelation between solutions, the notion of a neighbourhood is introduced based on a suitable distance measure. Thus a topology is constructed, producing a fitness landscape. A landscape is uniquely defined by the tuple $(S, N, f)$. If either the neighbourhood or the fitness function is changed, a different fitness landscape results, even if the set of solutions is maintained. Various landscape features can be extracted, such as the fitness distribution, the funnel structure, and the number and structure of optima.

Intuitively, a fitness landscape corresponds to a physical landscape, where the relative ‘heights’ of fitness values shape hills, plateaus, sinks and valleys. In the context of search algorithms, the fitness landscape can be formalised as a graph with solutions as its nodes and edges between neighbours [9]. The neighbourhood of a solution $s$ is the set of solutions obtained by applying a search operator, starting with $s$ [13]. In discrete space, as is the case with neural architectures, the operator may be defined as a reconfiguration of solution parameters, such as a bit flip in a binary string [10], [12]. In continuous space, the operator may be described using a distance metric, such as Euclidean distance [9].

In order to consider neural architecture space from the fitness landscape analysis perspective, definitions of candidate solutions, their fitness and neighbourhoods, as well as what constitutes a local minimum, must be provided. These are given in the subsections below.

A. Candidate solutions

The solution space $S$ considered in this study comprises all possible feedforward neural network architectures with a depth of up to 3 hidden layers, each comprising between 1 and 10 neurons, trained on a given data set. For ease of reference, neural networks with $n$ hidden layers are here simply referred to as having $n$ layers, while the full network invariably contains input and output layers. Only the network architecture is evaluated, randomly initialising weights and keeping other parameters, such as activation functions and their configurations, fixed. The neural architecture can be depicted as the set of 1-, 2- and 3-tuples of numbers 1 to 10. More formally, given $A = \{1, 2, \ldots, 10\}$, the solution space is parametrised as $S' = (A) \cup (A \times A) \cup (A \times A \times A)$.

The architecture search space is combinatorial, being discrete and finite. Within a defined maximum layer depth $d$ and individual layer width $w$, $|S| = \sum_{d=1}^{d} w^d$. With $d = 3$, $w = 10$ the total amounts to $|S| = |S'| = 1110$. $S$ is the set of feedforward networks parametrised as $S'$, which can be expressed as $S = \mathcal{X}(S')$, where $\mathcal{X}$ represents the derivation of the architecture instantiations corresponding to parametrised solutions in $S'$. Denote positional digits of $n$-tuple $s' \in S'$ by $(s'_1, \ldots, s'_d)$, a hidden layer containing $m$ neurons by $h[m]$ and a directional connection between layers by $h \rightarrow h$. Accordingly, a solution $s \in S$ can be constructed as

$$s = \mathcal{X}(s') = \mathcal{X}((s'_1, \ldots, s'_d)) = x \rightarrow h[s'_1] \rightarrow \cdots \rightarrow h[s'_d] \rightarrow y.$$  

For example, if $s' = (4, 3)$, then $s = \mathcal{X}((4, 3)) = x \rightarrow h[4] \rightarrow h[3] \rightarrow y$, as depicted in Fig. 1. The architecture space is fully enumerated by fitting all possible configurations within the defined bounds, training the corresponding model and evaluating its test performance.

B. Fitness evaluation

As the fitness value of a candidate solution, $f(s)$, we take the performance per configuration, computed as the R-squared ($R^2$) statistic for both regression and classification tasks, defined as

$$R^2 = 1 - \frac{\sum_{i}(y^{(i)} - \hat{y}^{(i)})^2}{\sum_{i}(y^{(i)} - \bar{y})^2},$$  

computed over $N$ predictions, where $y^{(i)}$ are the actual values of the response variable, $\hat{y}$ is the mean of the response values,
and \(y^{(i)}\) are the corresponding prediction values. \(R^2\) values typically range from a maximum of 1 for perfect predictions to 0, but exceptionally large errors can yield arbitrarily large negative values. \(R^2\) is selected due to its scale invariance, which aids interpretation of results. To facilitate the use of \(R^2\) in classification tasks, the uniform average for multiple outputs (target classes) is used per prediction to obtain residuals. A consistent fitness measure is, therefore, used throughout the study.

For each architecture, a batch of 30 models are trained and evaluated to control for variability in model performance due to random weight initialisation. The first model per batch is identically seeded, and subsequent training procedures correspondingly inherit from it. Thus the first 30 models trained to 0, but exceptionally large errors can yield arbitrarily large possible layer-wise neural decrements and increments applied to a solution. Let \(s \in S\) denote the set of all possible layer-wise neural decrements and increments applied to a solution \(s\). With \(W(s')\) as the set of offsets of the corresponding \(n\)-tuple \(s' \in S\), let \(N = \{1, \ldots, n\}\), width offsets are formalised by \(W(s) = W(X(s)) = \mathcal{X}(W'(s'))\), and

\[
W'(s') = D \cup I, \quad \text{where}
\]

\[
D = \{t' \in S' \mid \exists i \in N \text{ s.t. } t_i' = s_i' - 1, t_j' \neq s_j'\},
\]

\[
I = \{t' \in S' \mid \exists i \in N \text{ s.t. } t_i' = s_i' + 1, t_j' \neq s_j'\}.
\]

where \(t' \in S'\) implies that width bounds are not exceeded and the corresponding architecture must be a valid solution in \(S\), and \(\exists i\) denotes 'there exists a unique \(i\')'.

2) Depth Offsets: Depth is offset via either cloning or pruning. Cloning involves replicating the neuron count in an existing layer and adjacentely inserting an additional layer. Pruning is done by removing a selected layer, provided at least one layer remains. Due to pruning, neighbouring architectures are not necessarily symmetric, e.g. \((2, 2)\) is a neighbor of \((2, 2, 8)\) by virtue of pruning the third layer, but no reverse permutation is permitted. Let \(D(s)\) denote the set of all single layer cloning and pruning permutations to a solution \(s\), and \(D'(s')\) the corresponding set resulting from adding to or removing an entry from the \(n\)-tuple \(s'\). The depth offset operation is formalised as \(D(s) = D(X(s)) = \mathcal{X}(D'(s'))\), where

\[
D'(s') = \{t' \in S' \mid \exists i \in N \text{ s.t. } t_i' = s_i', t_{i-1}' = s_{i-1}' - 1, t_{i+1}' = s_{i+1}'\}.
\]

C. Neighbourhoods

Let the neighbourhood \(\mathcal{N}(s)\) of a solution \(s \in S\) be defined by an operation which either offsets the width of one layer by one or alternatively prunes or clones a layer, within allowed bounds. That is, the neighbourhood of solution \(s\) includes all arrangements with either the number of neurons incremented or decremented in one layer, or one layer duplicated or removed, without adjusting layer-wise neuron counts.

1) Width Offsets: Width offsets increment or decrement one layer’s neuron count. Let \(W(s)\) denote the set of all possible layer-wise neural decrements and increments applied to a solution \(s\). With \(W'(s')\) as the set of offsets of the corresponding \(n\)-tuple \(s' \in S\), let \(N = \{1, \ldots, n\}\), width offsets are formalised by \(W(s) = W(X(s)) = \mathcal{X}(W'(s'))\), and

\[
W'(s') = D \cup I, \quad \text{where}
\]

\[
D = \{t' \in S' \mid \exists i \in N \text{ s.t. } t_i' = s_i' - 1, t_j' \neq s_j'\},
\]

\[
I = \{t' \in S' \mid \exists i \in N \text{ s.t. } t_i' = s_i' + 1, t_j' \neq s_j'\}.
\]

The resulting set of \(n\)-tuples in \(S'\) must correspond to valid solutions in \(S\). Using Equation (2) and Equation (3), the neighbourhood of a solution \(s\) is defined as

\[
\mathcal{N}(s) = \mathcal{W}(s) \cup \mathcal{D}(s) = \mathcal{X}(W'(s')) \cup \mathcal{X}(D'(s')).
\]

Fig. 2 shows an example of the neighbourhood of the solution \(s' = (4, 3)\) (illustrated in Fig. 1), differentiating \(W(s)\) and \(D(s)\). The neighbourhood operation produces 5879 edges per landscape, and the average number of neighbours per solution\(^2\) is around 10.6.

D. Local Optima

A local optimum (LO) is a solution with a fitness value that matches or exceeds that of all solutions in its neighbourhood. Each local optimum \(i\) is surrounded by a basin of attraction, which is the set of solutions from which \(i\) will be reached by following a local search procedure. Formally, \(b_i = \{s \in S|h(s) = i\}\), and \(|b_i|\) is the cardinality of the basin. A best-improvement hill climbing local search is used in this study to locate local optima, defined in Algorithm 1.

Algorithm 1 Maximising best-improvement hill climbing

| Input: initial solution \(x\) |
| repeat |
| \(x' \leftarrow \arg \max_{y \in \mathcal{N}(x)} f(y)\) |
| if \(f(x') > f(x)\) then |
| \(x \leftarrow x'\) |
| end if |
| until \(x\) is a local optimum |

III. LOCAL OPTIMA NETWORKS

This study aimed to construct and investigate LONs for the neural architecture space. For each data set considered, the architecture space was fully enumerated, and the hill climbing terminus starting with each solution was determined, exhaustively populating LON nodes. To draw edges, the neighbourhood operation defined in Equation (4) was selected as a basis, applied by perturbation strength \(D = 2\), which is consistent with related literature [11], [12], and is also empirically found to be sufficient for escape, particularly from suboptimal solutions. An edge thus represents the relative likelihood of moving into a neighbouring basin of attraction from a local optimum after a controlled perturbation of up to two random moves. The value of \(D\) has a marked effect on the connectedness of the LON graph, since more random moves allow for farther reaching escape routes. Directed LON

\(^2\)every edge adds a neighbour to both solutions or nodes it connects, thus twice the edge count is used when computing the average: 5879(2)/1110
edges are weighted by the number of distinct paths resulting in a move between basins of the two local optima it connects. When calculating edge weights the edge counts between nodes are not scaled to a proportion of all distinct edges from the source node. Thus weights do not represent probabilities, but rather the relative likelihood of escape.

A. Funnels

In a monotonic LON (MLON) [12], non-deterioration is required for an edge to be drawn between nodes. MLONs yield the notion of funnels, which are the set of monotonic sequences leading to a particular local optimum. A funnel is a basin at the local optima level [14], with a sink node as its terminus. MLON sinks have out degree zero, and sources have in degree zero.

IV. NEURAL NETWORK CONFIGURATION

In the present context, the selection of hyper-parameter values as such is not as important as their consistency, to ensure commensurability of solutions between data sets. Apart from output activation and loss, all parameters were consistent between classification and regression tasks. All configuration settings used in the experiments are presented in Table I.

V. DATA

Models were trained on 9 different data sets: 6 classification tasks (Table II) and 3 regression tasks (III). While presenting a small sample, the intention was to analyse the fitness landscapes produced for a range of relatively common data sets. Classification sets included commonly used tabular and image recognition data. While these are relatively small data sets, the two categories differ significantly in kind and difficulty. All three regression sets are tabular, but have diverging input features and pattern counts.

With a limited model and data set size, each model was trained using a single CPU. Over an average of 50.86 epochs, the time required per model was an average of 30.77 seconds, ranging from 1.71 second for Iris to over 112 seconds for MNIST. Full enumeration of the solution space is, however, non-negligible. For each data set, each of the 1110 architectures was trained 30 times, totalling 33300 training procedures per set. To train the full compliment, a computing cluster was used to maximise the number of simultaneous training procedures. This consideration limits the number and size of the sets that can feasibly be included.

Image data arrays were flattened by concatenating the rows into a single input vector. Colour values were first converted to greyscale before being flattened. Numeric input variables were z-score normalised or standardised. Binary input variables were encoded as $-1$ and $1$, and multiple class inputs were one-hot encoded. Regression labels were left unscaled.

VI. RESULTS

Salient results relating to the extracted fitness landscapes are presented in Section VI-A, and local optima networks in Section VI-B. Data set names are abbreviated.

3https://www.chpc.ac.za/index.php/resources/lengau-cluster
TABLE II
Classification Data Sets

| NAME          | TYPE | PATTERNS | INPUTS | CLS |
|---------------|------|----------|--------|-----|
| IRIS [15]     | TAB  | 150      | 4      | 3   |
| WINE [16]     | TAB  | 178      | 13     | 3   |
| MNIST [17]    | IMG  | 70000    | 28 x 28| 10  |
| F. MNIST [18] | IMG  | 70000    | 28 x 28| 10  |
| CIFAR-10 [19] | IMG  | 60000    | 32 x 32| 10  |
| CIFAR-100 [19]| IMG  | 60000    | 32 x 32| 100 |

TABLE III
Regression Data Sets

| NAME            | TYPE    | PATTERNS | INPUTS |
|-----------------|---------|----------|--------|
| WINE QUALITY [20]| TABULAR | 4898     | 11     |
| INSURANCE [21]  | TABULAR | 1338     | 6      |
| HOUSING [22]    | TABULAR | 506      | 13     |

A. Fitness Landscapes

A bird's eye view of the fitness values produced by the trained neural networks, as represented by the box plots in Fig. 3, reveals a wide range of outcomes across the different data sets. Only iris and wine have solutions that achieved an $R^2$ score of 1. Fitness values varied significantly between data sets in terms of mean and median statistics, range and distribution.

For all data sets, the bulk of the fitness values are bounded by 0 and 1, but all sets exhibited solutions with negative fitness, representing a complete failure to fit the response data. Negative outliers were much more extreme for the regression tasks, in particular insur and housing. Softmax, used for classification output, produced a probability distribution of values between 0 and 1 for all classes, summed to a total value of 1, effectively bounding the numeric divergence possible when the target is numeric and of arbitrary scale, as in this study.

B. Local Optima Networks

A LON was extracted for each data set, with local optima (LO) found using best-improvement hill climbing as the nodes and edges drawn between LO if any controlled perturbation to one local optimum (source) reached a solution within the basin of another local optimum (target), as described in Section III. Table IV provides a comparison of basic features describing the resulting networks, further illustrating the variation between fitness landscapes.

1) Local Optima Fitness Distribution: A comparison of LO fitnesses is displayed in Fig. 4, demonstrating an array of model capacities relative to the provided data sets. The distributions of LO fitness values also bares resemblance to the distributions of the solution space as a whole (Fig. 3). The choice of $R^2$ as the fitness metric resulted in all solution fitnesses being globally unique. Consequentially, all data sets produced a unique global optimum (Table IV).

2) Network Modality: The number of local optima varied from as few as 4 for mnist to as many as 101 for cifar100 – more than 9% of the search space. As can be expected, the edge count correlated with the number of nodes that they connect, as demonstrated in Fig. 5. While the maximum number of potential directional connections between nodes increased exponentially as the number of nodes increased,
totalling $n(n - 1)$ for $n$ nodes, it was observed that for the selected data sets, the edge count increased in a linear fashion for networks with more than 20 nodes, approximated by $35n - 600$. A possible explanation is that higher modality implies a higher number of smaller basins, so that on average a higher proportion of basins are out of reach from any particular solution.

A lower global optimum value reduced the fitness range. This is correlated with an increase in modality when comparing the *cifar* and *mnist* data sets, but is not consistent, as illustrated in Fig. 5. A better indicator may be the fitness standard deviation (SD) between local optima. Fig. 6 depicts the number of network nodes over the fitness SD per data set. At the extremes, *cifar100* had the lowest SD and the highest modality, while *mnist* expressed the inverse. The allocation of the intermediate cases made the overall trajectory resemble exponential decay, with the node and edge count decreasing rapidly with increases near the lower end of the SD range. Yet there was a wide spread, for example, between *insur* and *wine*, with similar SD. For the data sets considered, the inverse relation between SD and node count was thus found to be clearest at the extremes.

Since the LON edges are directed, the breakdown of improving and deteriorating connections can be compared, that is, whether following an edge results in a move to a local optimum with a better or worse fitness. On average, improving moves were found to be around 1.8 times more common.

3) Local Optima Basins: A predominantly improving edge set makes sense when considering the effect of relative basin sizes between solutions. Fitter solutions can generally be expected to have bigger basins, easier to reach, but harder to escape from. Fig. 7 provides an overview of the relative LO basin sizes across fitness values per data set. Overall, relative basin size can be observed to scale up as fitness increases. The most extreme example is *fmnist*, which has an absolutely dominant basin at its global optimum.

Sizeable suboptimal basins also exist, sometimes exceeding the basin size at the global optimum. Fig. 8 plots the 2-dimensional LON graph for *iris*, with self loops removed to simplify the layout. The size of each vertex is determined by the incoming strength of LON edges, demonstrating the correspondence between basin size and likelihood of being reached by a perturbation. For the same reason that larger basins at higher fitness levels make improving moves more likely, the basins of suboptimal solutions may provide obstacles to optimisation over the fitness landscape, even forming suboptimal sinks. Despite the prominence of this solution in
the LON, it is not a sink, and is outward connected to the comparatively modest global optimum.

4) Funnels: Remarkably, the only data set producing a suboptimal funnel was cifar10, depicted in Fig. 9. The density of the graph is a result of the high modality pointed out in Section VI-B2. Despite the range of the fitness landscapes that have been populated, therefore, each contains a simple funnel structure and is therefore suitable for local search optimisation. This is a significant indication of the potential effectiveness of using LONs for architecture optimisation.

5) Local Optimisation: As an indication of fitness landscape searchability, the performance of an iterated local search (ILS) procedure, described in Algorithm 2, was evaluated. The perturbation is defined as the same operation used to draw LON edges (Section III), with perturbation strength $k = 2$ and stopping criterion $t = 20$. Aggregating the results of 100 runs per data set, ILS discovered a global top 5 solution after evaluating on average only 50.67 evaluations (the average of the median evaluations needed per data set). Fig. 10 provides a comparison of first discoveries of near optimal solutions. For almost all data sets, the median first encounter with a top 5 solution is within 50 evaluations, while the comparative hardness of cifar10 is clear. Each run only reached a fraction of all solutions before exhausting escape possibilities. Despite the limited reach, 65% of overall runs produced the global optimum, after an average of 92.56 evaluations, which is less than a tenth of the solution space.

VII. CONCLUSION

This study presented an initial evaluation of the feasibility of using local optima network (LON) analysis to characterise and optimise over the feedforward neural network (FFNN) architecture space. The solution space was defined in such a way that full enumeration using multiple data sets was feasible. One of the main findings is the relatively simple global funnel structures of the resulting monotonic LONs (MLONs).

The LONs produced for the data sets considered in this study exhibited diverging characteristics in terms of their modality, fitness distribution, basin size distribution and global optimum incoming strength. All but one MLON, however, had a single global funnel, with only two funnels in the remaining local optima network. The simple funnel structure supports the use of relatively simple local optimisers like iterative local search as opposed to significantly more computationally expensive population-based methods.

A high standard deviation between local optimum fitness values was found to be related to low modality. Low modality in turn improves the relative ease of fitness optimisation. When encountered at low fitness levels, a very low standard deviation may indicate that the model is not sufficiently
powerful for the given task, as is the case with the CIFAR-10 and CIFAR-100 data sets.

One line of extension for future work is to investigate larger FFNN architecture spaces more representative of real-world tasks. Larger models are likely to improve overall fitness, which was found to correspond to simpler LONs and improve sampling performance. Another extension is to study LONs produced using different architectural arrangements, like recurrent or convolutional neural networks. Feedforward neural networks have insufficient representational capacity for many real-world problems, making basic network augmentation commonplace in modern machine learning.

This study focused exclusively on the architecture space, but weight initialisation is also known to play an important role in model performance [23], [24]. To investigate the weight initialisation space, some form of discretisation is required. A promising option, considering the relatively low number of possible configurations, is to study combinations of initial weight signs, keeping the magnitude constant [24]. This line of investigation may contribute towards understanding what makes lottery tickets successful [8].

ACKNOWLEDGEMENT

The authors would like to thank the Centre for High Performance Computing (CHPC) (http://chpc.ac.za) for the use of their cluster to obtain the data for this study.

REFERENCES

[1] R. Caruana, S. Lawrence, and C. L. Giles, “Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping,” in Advances in neural information processing systems, 2001, pp. 402–408.

[2] L. Sagun, V. U. Guney, G. B. Arous, and Y. LeCun, “Explorations on high dimensional landscapes,” arXiv preprint arXiv:1412.6615, 2014.

[3] G. Cybenko, “Approximation by superpositions of a sigmoidal function,” Mathematics of control, signals and systems, vol. 2, no. 4, pp. 303–314, 1989.

[4] K. Hornik, M. Stinchcombe, H. White et al., “Multilayer feedforward networks are universal approximators,” Neural networks, vol. 2, no. 5, pp. 359–366, 1989.

[5] G. Cybenko, “Approximation by superpositions of a sigmoidal function,” Mathematics of control, signals and systems, vol. 2, no. 4, pp. 303–314, 1989.

[6] A. S. Bosman, A. P. Engelbrecht, and M. Helbig, “Loss surface modality of feed-forward neural network architectures,” in 2020 International Joint Conference on Neural Networks (IJCNN). IEEE, 2020, pp. 1–8.

[7] J. M. Springer and G. T. Kenyon, “It’s hard for neural networks to learn the game of life,” arXiv preprint arXiv:2009.01398, 2020.

[8] J. Frankle and M. Carbin, “The lottery ticket hypothesis: Finding sparse, trainable neural networks,” arXiv preprint arXiv:1803.03635, 2018.

[9] K. M. Malan and A. P. Engelbrecht, “A survey of techniques for characterising fitness landscapes and some possible ways forward,” Information Sciences, vol. 241, pp. 148–163, 2013.

[10] G. Ochoa, M. Tomassini, S. Vérel, and C. Darabos, “A study of NK landscapes’ basins and local optima networks,” in Proceedings of the 10th annual conference on Genetic and evolutionary computation, 2008, pp. 555–562.

[11] W. Mostert, K. M. Malan, G. Ochoa, and A. P. Engelbrecht, “Insights into the feature selection problem using local optima networks,” in European Conference on Evolutionary Computation in Combinatorial Optimization (Part of EvoStar). Springer, 2019, pp. 147–162.

[12] G. Ochoa, N. Veerapen, F. Daolio, and M. Tomassini, “Understanding phase transitions with local optima networks: number partitioning as a case study,” in European Conference on Evolutionary Computation in Combinatorial Optimization. Springer, 2017, pp. 233–248.

[13] I. Moser, M. Gheorghita, and A. Aleti, “Identifying features of fitness landscapes and relating them to problem difficulty.” Evolutionary computation, vol. 25, no. 3, pp. 407–437, 2017.

[14] S. L. Thomson, G. Ochoa, S. Verel, and N. Veerapen, “Inferring future landscapes: sampling the local optima level.” Evolutionary computation, vol. 28, no. 4, pp. 612–641, 2020.

[15] R. A. Fisher, “The use of multiple measurements in taxonomic problems,” Annals of eugenics, vol. 7, no. 2, pp. 179–188, 1936.

[16] M. Forina, R. Leardi, C. Armanino, S. Lanteri, P. Conti, and P. Princi, “Parvus: An extendable package of programs for data exploration, classification and correlation,” Journal of Chemometrics, vol. 4, no. 2, pp. 191–193, 1988.

[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

[18] H. Xiao, K. Rasul, and R. Vollgraf, “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms,” arXiv preprint arXiv:1708.07747, 2017.

[19] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in Proceedings of the IEEE international conference on computer vision, 2017, pp. 1026–1034.

[20] J. Frankle and M. Carbin, “The lottery ticket hypothesis: Finding sparse, trainable neural networks,” arXiv preprint arXiv:1803.03635, 2018.

[21] A. P. Engelbrecht, “Loss surface modality of feed-forward neural network architectures,” in 2020 International Joint Conference on Neural Networks (IJCNN). IEEE, 2020, pp. 1–8.

[22] J. M. Springer and G. T. Kenyon, “It’s hard for neural networks to learn the game of life,” arXiv preprint arXiv:2009.01398, 2020.

[23] J. Frankle and M. Carbin, “The lottery ticket hypothesis: Finding sparse, trainable neural networks,” arXiv preprint arXiv:1803.03635, 2018.

[24] M. Forina, R. Leardi, C. Armanino, S. Lanteri, P. Conti, and P. Princi, “Parvus: An extendable package of programs for data exploration, classification and correlation,” Journal of Chemometrics, vol. 4, no. 2, pp. 191–193, 1988.

[25] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

[26] H. Xiao, K. Rasul, and R. Vollgraf, “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms,” arXiv preprint arXiv:1708.07747, 2017.

[27] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in Proceedings of the IEEE international conference on computer vision, 2017, pp. 1026–1034.

[28] J. Frankle and M. Carbin, “The lottery ticket hypothesis: Finding sparse, trainable neural networks,” arXiv preprint arXiv:1803.03635, 2018.