Losslessly Compressible Neural Network Parameters

Matthew Farrugia-Roberts
School of Computing and Information Systems
The University of Melbourne
matthew@far.in.net

Abstract

To better understand complexity in neural networks, we theoretically investigate the idealised phenomenon of lossless network compressibility, whereby an identical function can be implemented with fewer hidden units. In the setting of single-hidden-layer hyperbolic tangent networks, we define the rank of a parameter as the minimum number of hidden units required to implement the same function. We give efficient formal algorithms for optimal lossless compression and computing the rank of a parameter. We also characterise the set of parameters with a given maximum rank as a union of linear subspaces. The lossless compression operations we study have implications for the approximate compressibility of nearby parameters and parameters in more complex architectures.

1 Introduction

Learned neural networks are often simpler than parameter counting would suggest. Architectures used in practice can easily memorise randomly labelled data (Zhang et al., 2017, 2021). Yet, they tend to learn simple functions that are *approximately compressible*, in that there are smaller networks implementing similar functions (e.g., Buciluǎ et al., 2006; Hinton et al., 2014; Sanh et al., 2019).

To advance our understanding of neural network complexity, we propose studying the idealised phenomenon of *lossless compressibility* of neural network parameters, whereby an *identical* function can be implemented by some network with fewer units.

In this paper, we study losslessly compressible parameters in the setting of single-hidden-layer hyperbolic tangent networks. While this setting is not immediately relevant to modern deep learning, our analysis applies partially to a much broader class of nonlinearities and to any individual feed-forward layer of a larger architecture. We therefore offer the following theoretical contributions as a first step towards understanding lossless compressibility in modern architectures.

1. In Section 4, we give efficient formal algorithms for optimal lossless compression of single-hidden-layer hyperbolic tangent networks, and for computing the *rank* of a parameter—the minimum number of hidden units required to implement the same function.
2. In Section 5, we invert our optimal lossless compression algorithm to offer a characterisation of the subset of the parameter space containing parameters with a given maximum rank as a union of linear subspaces.

The study of lossless compressibility could offer a new lens on the phenomenon of approximate compressibility in deep learning. The neighbours of losslessly compressible parameters implement similar functions, and are approximately compressible using the same compression operations. We hypothesise that the phenomenon of approximate compressibility in deep learning practice is partially explained by such operations. If so, studying lossless compressibility could reveal new approximate compression operations that can help us to understand the complexity of learned neural networks. We discuss this research direction in Section 6.

Machine Learning and Compression Workshop @ NeurIPS 2024.

2 Related work

Approximate compression. There is a sizeable empirical literature on approximate compression in neural networks, including via pruning, quantisation, and distillation (see Cheng et al., 2018, 2020 or Choudhary et al., 2020 for an overview). Approximate compressibility has also been proposed as a learning objective (Hinton and van Camp, 1993; Aytekin et al., 2019) and used in deriving generalisation bounds (Suzuki et al., 2020a,b). We adopt a convention of measuring network size by counting units, noting that this is one of various alternative conventions in the literature, such as counting weights or measuring the description length of weights.

Lossless compression. There has been less work on *lossless* compression, requiring the network's outputs to be exactly preserved. Serra et al. (2020) give a partial lossless compression algorithm for multi-layer ReLU networks that preserves the implemented function over some input domain. Their algorithm exploits *some* opportunities for removing units, but does not claim to (and does not in general) *optimally* compress networks. By contrast, we give an algorithm for optimal lossless compression over all inputs in the simpler setting of single-hidden-layer hyperbolic tangent networks.

Functional equivalence. For single-hidden-layer hyperbolic tangent networks, Sussmann (1992) showed that, for almost all parameters, two parameters implement identical functions if and only if they are related by simple operations of exchanging or negating the weights of hidden units. Similar results have been shown for various architectures, including architectures with different nonlinearities (Albertini et al., 1993; Kůrková and Kainen, 1994; Phuong and Lampert, 2020), multiple hidden layers (Chen et al., 1993; Fefferman and Markel, 1993; Fefferman, 1994; Phuong and Lampert, 2020), and more complex connection graphs (Vlačić and Bölcskei, 2021, 2022).

Lossless compressibility is precisely the existence of functionally equivalent parameters with fewer units. The function-preserving operations cited above generally preserve the number of units.[1] Farrugia-Roberts (2023) studies additional function-preserving operations available for losslessly compressible single-hidden-layer hyperbolic tangent parameters, giving an algorithm for identifying pairs of functionally equivalent parameters that achieves optimal lossless compression as a side-effect. We give a more efficient optimal lossless compression algorithm.

Lossless expansion. There is also work adopting a dual perspective of cataloguing various ways of *adding* hidden units to a neural network while exactly preserving the implemented function. Fukumizu and Amari (2000) and Fukumizu et al. (2019) show that some of the resulting losslessly compressible parameters are critical points of the loss landscape. Şimşek et al. (2021) and Farrugia-Roberts (2023) show that sets of equivalent losslessly compressible parameters have a rich structure that reaches throughout the parameter space. We show a similar result for the set of all sufficiently losslessly compressible (not necessarily equivalent) parameters in Section 5.

Information singularities. Losslessly compressible parameters are singularities in the Fisher information landscape (Fukumizu, 1996), and if they are critical points of the loss landscape they are degenerate. This makes them highly relevant to singular statistical theories of deep learning (Watanabe, 2009; Wei et al., 2023). For example, these singularities influence learning dynamics in their neighbourhood (Amari et al., 2006; Wei et al., 2008; Cousseau et al., 2008; Amari et al., 2018).

3 Preliminaries

Architecture. We consider a family of fully-connected, feed-forward neural network architectures with one input unit, one biased output unit, and one hidden layer of $h \in \mathbb{N}$ biased hidden units with the hyperbolic tangent nonlinearity $\tanh(z) = (e^z - e^{-z})/(e^z + e^{-z})$. The weights and biases of the network are encoded in a parameter vector in the format $w = (a_1, b_1, c_1, \ldots, a_h, b_h, c_h, d) \in \mathcal{W}_h = \mathbb{R}^{3h+1}$, where for each hidden unit $i = 1, \ldots, h$ there is an *outgoing weight* $a_i \in \mathbb{R}$, an *incoming weight* $b_i \in \mathbb{R}$, and a *bias* $c_i \in \mathbb{R}$; and $d \in \mathbb{R}$ is the *output unit bias*. Thus each parameter $w \in \mathcal{W}_h$ indexes a mathematical function $f_w : \mathbb{R} \to \mathbb{R}$ such that $f_w(x) = d + \sum_{i=1}^{h} a_i \tanh(b_i x + c_i)$. All of our results generalise to networks with multi-dimensional inputs and outputs (see Appendix A).

[1] It follows that losslessly compressible parameters occupy a measure zero subset of parameter space. We note that learning exerts a non-random selection pressure, so these parameters may still be relevant in practice.

Reducibility. Two parameters $w \in \mathcal{W}_h$ and $w' \in \mathcal{W}_{h'}$ are *functionally equivalent* if $f_w = f_{w'}$ as functions on $\mathbb{R}$ ($\forall x \in \mathbb{R}, f_w(x) = f_{w'}(x)$). A parameter $w \in \mathcal{W}_h$ is *(losslessly) compressible* if and only if w is functionally equivalent to some $w' \in \mathcal{W}_{h'}$ with fewer hidden units $h' < h$ (otherwise, w is *incompressible*). Sussmann (1992) showed that a simple condition, *reducibility,* is necessary and sufficient for lossless compressibility. A parameter $(a_1, b_1, c_1, \ldots, a_h, b_h, c_h, d) \in \mathcal{W}_h$ is *reducible* if and only if it satisfies any of the following *reducibility conditions*:

(i) $a_i = 0$ for some i, or

(ii) $b_i = 0$ for some i, or

(iii) $(b_i, c_i) = (b_j, c_j)$ for some $i \neq j$, or

(iv) $(b_i, c_i) = (-b_j, -c_j)$ for some $i \neq j$.

Each reducibility condition suggests a simple operation to remove a hidden unit while preserving the function (Sussmann, 1992; Farrugia-Roberts, 2023).

(i) Units with zero outgoing weight contribute zero to the function, and can be *eliminated*, that is, can be removed from the network.

(ii) Units with zero incoming weight contribute a constant to the function, and can also be *eliminated* after incorporating the constant into the output bias.

(iii) Unit pairs with identical incoming weight and bias contribute proportionally to the function, and can be *merged* into a single unit with the sum of their outgoing weights as the new outgoing weight (for a net removal of one unit).

(iv) Unit pairs with identically negative incoming weight and bias also contribute in proportion, since the hyperbolic tangent is odd. Such pairs can also be *merged* into a single unit with the difference of their outgoing weights as the new outgoing weight.

4 Algorithms for optimal lossless compression and rank

We consider the problem of lossless neural network compression: finding, given a compressible parameter, a functionally equivalent but incompressible parameter. The following algorithm solves this problem by eliminating units meeting reducibility conditions (i) or (ii) and merging unit pairs meeting reducibility conditions (iii) or (iv) in ways preserving functional equivalence.

Algorithm 4.1 (Lossless neural network compression)**.** Given $h \in \mathbb{N}$, proceed:

procedure COMPRESS($w = (a_1, b_1, c_1, \ldots, a_h, b_h, c_h, d) \in \mathcal{W}_h$)
  ▷ *Stage 1: Eliminate units with incoming weight zero (incorporate into new output bias δ)* ◁
  $I \leftarrow \{\, i \in \{1, \ldots, h\} \,:\, b_i \neq 0 \,\}$
  $\delta \leftarrow d + \sum_{i \notin I} \tanh(c_i) \cdot a_i$
  ▷ *Stage 2: Partition and merge remaining units by incoming weight and bias* ◁
  $\Pi_1, \ldots, \Pi_J \leftarrow$ partition I by the value of $\operatorname{sign}(b_i) \cdot (b_i, c_i)$
  for $j \leftarrow 1, \ldots, J$ **do**
    $\alpha_j \leftarrow \sum_{i \in \Pi_j} \operatorname{sign}(b_i) \cdot a_i$
    $\beta_j, \gamma_j \leftarrow \operatorname{sign}(b_{\min \Pi_j}) \cdot (b_{\min \Pi_j}, c_{\min \Pi_j})$
  end for
  ▷ *Stage 3: Eliminate merged units with outgoing weight zero* ◁
  $k_1, \ldots, k_r \leftarrow \{\, j \in \{1, \ldots, J\} \,:\, \alpha_j \neq 0 \,\}$
  ▷ *Construct a new parameter with the remaining merged units* ◁
  return $(\alpha_{k_1}, \beta_{k_1}, \gamma_{k_1}, \ldots, \alpha_{k_r}, \beta_{k_r}, \gamma_{k_r}, \delta) \in \mathcal{W}_r$
end procedure

Theorem 4.1 (Algorithm 4.1 correctness)**.** *Given $w \in \mathcal{W}_h$, compute $w' = \text{COMPRESS}(w) \in \mathcal{W}_r$. Then*

(i) $f_{w'} = f_w$, and

(ii) w' is incompressible.

Proof. (i): Observe that units eliminated in Stage 1 contribute a constant, units merged in Stage 2 have proportional contributions, and merged units eliminated in Stage 3 do not contribute. Thus following the steps of the algorithm we rearrange the summation defining f_w to have the form of $f_{w'}$. For each $x \in \mathbb{R}^n$,

$$\begin{aligned}
f_w(x) &= d + \sum_{i=1}^{h} a_i \tanh(b_i x + c_i) \\
&= d + \sum_{i \notin I} a_i \tanh(c_i) + \sum_{i \in I} a_i \tanh(b_i x + c_i) && \text{(cf. line 3)} \\
&= \delta + \sum_{i \in I} a_i \tanh(b_i x + c_i) && \text{(cf. line 4)} \\
&= \delta + \sum_{j=1}^{J} \sum_{i \in \Pi_j} a_i \tanh(b_i x + c_i) && \text{(cf. line 6)} \\
&= \delta + \sum_{j=1}^{J} \sum_{i \in \Pi_j} \operatorname{sign}(b_i) \cdot a_i \tanh(\operatorname{sign}(b_i) \cdot b_i x + \operatorname{sign}(b_i) \cdot c_i) && \text{(tanh odd)} \\
&= \delta + \sum_{j=1}^{J} \left(\sum_{i \in \Pi_j} \operatorname{sign}(b_i) \cdot a_i \right) \tanh(\beta_j x + \gamma_j) && \text{(cf. lines 6, 9)} \\
&= \delta + \sum_{j=1}^{J} \alpha_j \tanh(\beta_j x + \gamma_j) && \text{(cf. line 8)} \\
&= \delta + \sum_{j=1}^{r} \alpha_{k_j} \tanh(\beta_{k_j} x + \gamma_{k_j}) && \text{(cf. line 12)} \\
&= f_{w'}(x). && \text{(cf. line 14)}
\end{aligned}$$

(ii): By construction, $w' \in \mathcal{W}_r$ fails to satisfy each of the reducibility conditions: (i) no α_k is zero, due to line 12; (ii) no β_k is zero, due chiefly to line 3; (iii), (iv) all $\pm(\beta_k, \gamma_k)$ are distinct, due chiefly to line 6. So w' is not reducible and is thus incompressible by Sussmann (1992). □

We define the *rank*[2] of a neural network parameter $w \in \mathcal{W}_h$, denoted $\operatorname{rank}(w)$, as the minimum number of hidden units required to implement f_w: $\operatorname{rank}(w) = \min \{ h' \in \mathbb{N} : \exists w' \in \mathcal{W}_{h'};\ f_w = f_{w'} \}$. The rank is also the number of hidden units in $\textsc{Compress}(w)$, since Algorithm 4.1 produces an incompressible parameter. Computing the rank is therefore a trivial matter of counting the units after performing lossless compression. The following is a streamlined algorithm, following Algorithm 4.1 but removing steps that don't influence the final count.

Algorithm 4.2 (Rank of a neural network parameter)**.** Given $h \in \mathbb{N}$, proceed:

procedure $\textsc{Rank}(w = (a_1, b_1, c_1, \ldots, a_h, b_h, c_h, d) \in \mathcal{W}_h)$
  ▷ *Stage 1: Identify units with incoming weight nonzero (not eliminated)* ◁
  $I \leftarrow \{ i \in \{1, \ldots, h\} : b_i \neq 0 \}$
  ▷ *Stage 2: Partition these units and compute outgoing weights for would-be merged units* ◁
  $\Pi_1, \ldots, \Pi_J \leftarrow$ partition I by the value of $\operatorname{sign}(b_i) \cdot (b_i, c_i)$
  $\alpha_j \leftarrow \sum_{i \in \Pi_j} \operatorname{sign}(b_i) \cdot a_i$ **for** $j \leftarrow 1, \ldots, J$
  ▷ *Stage 3: Count merged units with outgoing weight nonzero* ◁
  return $|\{ j \in \{1, \ldots, J\} : \alpha_j \neq 0 \}|$ ▷ $|S|$ *denotes set cardinality*
end procedure

Theorem 4.2 (Algorithm 4.2 correctness)**.** *Given* $w \in \mathcal{W}_h$, $\operatorname{rank}(w) = \textsc{Rank}(w)$.

Proof. Let r be the number of hidden units in $\textsc{Compress}(w)$. Then $r = \operatorname{rank}(w)$ by Theorem 4.1. Moreover, comparing Algorithms 4.1 and 4.2, observe $\textsc{Rank}(w) = r$. □

[2] In the multi-dimensional case (see Appendix A), our notion of rank generalises the familiar notion from linear algebra, where the rank of a linear transformation corresponds to the minimum number of hidden units required to implement the transformation with an unbiased linear neural network (cf. Piziak and Odell, 1999). Unlike in the linear case, our non-linear rank is not bounded by the input or output dimensionality.

5 A characterisation of the class of bounded-rank parameters

The reducibility conditions characterise the set of parameters $w \in \mathcal{W}_h$ with $\operatorname{rank}(w) \leq h-1$. In this section we characterise the set of parameters with an arbitrary rank bound.

Let $r, h \in \mathbb{N}$ with $r \leq h$. The *bounded rank region* of rank r is the subset of parameters of rank at most r, $\mathfrak{B}_r = \{\, w \in \mathcal{W}_h \,:\, \operatorname{rank}(w) \leq r \,\} \subseteq \mathcal{W}_h$. The key to characterising bounded rank regions is that for each parameter in $\mathfrak{B}_r$, at least $h-r$ units would be removed during lossless compression. Considering the various possible ways in which units can be removed in the course of Algorithm 4.1 leads to a characterisation of the bounded rank region as a union of linear subspaces.

To this end, let $H = \{1, \ldots, h\}$, and define a *compression trace on h units* as a 4-tuple $(\bar{I}, \Pi, \bar{K}, \sigma)$ where $\bar{I} \subseteq H$ is a subset of units (conceptually, those to be removed in Stage 1), $\Pi = \Pi_1, \ldots, \Pi_J$ is a partition of $H \setminus \bar{I}$ (the remaining units) into J groups (to be merged in Stage 2), $\bar{K} \subseteq \{1, \ldots, J\}$ (merged units removed in Stage 3), and $\sigma \in \{-1, +1\}^h$ is a sign vector (unit orientations for purposes of merging). The length of the compression trace $(\bar{I}, \Pi, \bar{K}, \sigma)$ on h units is $J - \left|\bar{K}\right|$ (representing the number of units remaining). A compression trace of length r thus captures the notion of a "way in which $h-r$ units can be removed in the course of Algorithm 4.1".

Theorem 5.1. *Let $r \leq h \in \mathbb{N}$. The bounded rank region $\mathfrak{B}_r \subseteq \mathcal{W}_h$ is a union of linear subspaces*

$$\mathfrak{B}_r = \bigcup_{(\bar{I}, \Pi, \bar{K}, \sigma) \in \Xi(h,r)} \left(\bigcap_{i \in \bar{I}} S_i^{(1)} \cap \bigcap_{j=1}^{J} S_{\Pi_j, \sigma}^{(2)} \cap \bigcap_{k \in \bar{K}} S_{\Pi_k, \sigma}^{(3)} \right) \tag{1}$$

where $\Xi(h, r)$ denotes the set of all compression traces on h units with length r;

$$S_i^{(1)} = \{\, (a_1, b_1, c_1, \ldots, a_h, b_h, c_h, d) \in \mathcal{W}_h \,:\, b_i = 0 \,\};$$

$$S_{\Pi,\sigma}^{(2)} = \{\, (a_1, b_1, c_1, \ldots, a_h, b_h, c_h, d) \in \mathcal{W}_h \,:\, \forall i, j \in \Pi,\ \sigma_i b_i = \sigma_j b_j \wedge \sigma_i c_i = \sigma_j c_j \,\};\ \textit{and}$$

$$S_{\Pi,\sigma}^{(3)} = \left\{\, (a_1, b_1, c_1, \ldots, a_h, b_h, c_h, d) \in \mathcal{W}_h \,:\, \textstyle\sum_{i \in \Pi} \sigma_i a_i = 0 \,\right\}.$$

Proof. ($\supseteq$): Suppose $w = (a_1, b_1, c_1, \ldots, a_h, b_h, c_h, d) \in \mathcal{W}_h$ is in the union in (1), and therefore in the intersection for some compression trace $(\bar{I}, \Pi, \bar{K}, \sigma)$. The constraints imposed on w by membership in this intersection imply that the network is compressible:

1. For $i \in \bar{I}$, since $w \in S_i^{(1)}$, $b_i = 0$, so unit i can be removed.
2. For $j = 1, \ldots, J$, since $w \in S_{\Pi_j,\sigma}^{(2)}$, the units in Π_j can be merged together.
3. For $k \in \bar{K}$, since $w \in S_{\Pi_k,\sigma}^{(3)}$, merged unit k has outgoing weight 0 and can be removed.

It follows that there is a parameter with $J - \left|\bar{K}\right|$ units that is functionally equivalent to w. Therefore $\operatorname{rank}(w) \leq J - \left|\bar{K}\right| = r$ and $w \in \mathfrak{B}_r$.

($\subseteq$): Conversely, suppose $w \in \mathfrak{B}_r$. Construct a compression trace following $\textsc{Compress}(w)$. First, set $\sigma_i = \operatorname{sign}(b_i)$ where $b_i \neq 0$ (if $b_i = 0$, set $\sigma_i = \pm 1$ arbitrarily, this has no effect). Then:

1. Set $\bar{I} = \{1, \ldots, h\} \setminus I$ where I is computed on line 3. It follows that for $i \in \bar{I}$, $w \in S_i^{(1)}$.
2. Set Π to the partition computed on line 6. It follows that for $j = 1, \ldots, J$, $w \in S_{\Pi_j,\sigma}^{(2)}$.
3. Set $\bar{K} = \{\, j \in \{1, \ldots, J\} \,:\, \alpha_j = 0 \,\}$ (cf. lines 8,12). Thus for $k \in \bar{K}$, $w \in S_{\Pi_k,\sigma}^{(3)}$.

By construction w is in $\bigcap_{i \in \bar{I}} S_i^{(1)} \cap \bigcap_{j=1}^{J} S_{\Pi_j,\sigma}^{(2)} \cap \bigcap_{k \in \bar{K}} S_{\Pi_k,\sigma}^{(3)}$. However, the compression trace $(\bar{I}, \Pi, \bar{K}, \sigma)$ has length $\operatorname{rank}(w) \leq r$. If $\operatorname{rank}(w) < r$, remove constraints on $r - \operatorname{rank}(w)$ units by some combination of the following operations: (1) remove one unit from $\bar{I}$ (add it as singleton group to Π), (2) remove one unit from a non-singleton group in Π (add it back to Π as a singleton group), and/or (3) remove one merged unit from $\bar{K}$. None of these operations add non-trivial constraints on w, so it's still the case that w is in the intersection for the modified compression trace. However, now the length of the compression trace is r, so it follows that w is in the union as required. □

6 Discussion

We have developed an algorithmic framework for lossless compressibility in single-hidden-layer hyperbolic tangent networks. The *rank* measures a parameter's lossless compressibility. Section 4 offers efficient algorithms for performing optimal lossless compression and computing the rank, and Section 5 inverts these algorithms to characterise the set of parameters of bounded rank.

While losslessly compressible parameters are themselves atypical, being *near* a losslessly compressible parameter is a sufficient condition for a parameter to be *approximately* compressible, since similar parameters implement similar functions. Moreover, such parameters may be approximately compressed using the very same unit elimination or merging operations as would be used to compress the nearby losslessly compressible parameter.

It's possible that this specific kind of approximate compressibility—compressibility explained by proximity to losslessly compressible parameters—accounts for at least part of the approximate compressibility observed in deep learning practice. If so, studying losslessly compressible parameters as in this work could lead to a better understanding of the phenomenon of approximate compressibility in deep learning. We conclude by discussing important questions for future theoretical and empirical work in this direction.

Does deep learning find parameters with low-rank neighbours in practice? While low proximate rank implies approximate compressibility, the converse is false. In particular, approximate compression approaches often require the implemented function to be similar only for certain inputs. Moreover, there are typically ways of implementing similar functions with dissimilar parameters (Petersen et al., 2021). It is an open question to what extent the approximate compressibility observed in deep learning practice coincides with proximity to losslessly compressible parameters.

There is evidence that lossless compressibility is relevant in at least some cases. While investigating the structure of learned neural networks, Casper et al. (2021) found many units with weak or correlated outputs, and found that these units could be removed without a large effect on performance, using elimination and merging operations bearing a striking resemblance to Sussmann's reducibility operations (Section 3). Deeply studying this phenomenon and other similar empirical questions suggested by the theory of lossless compressibility is an important direction for future work.

As a prerequisite for this work, we need methods for detecting proximity to losslessly compressible parameters. Computing the rank of the lowest-rank parameter in a given neighbourhood is a combinatorial optimisation problem that is computationally intractible in the worst case. However, it seems unlikely that the parameters encountered in practice will be pathological, and efficient approximate detection methods will suffice in practice.

What additional opportunities for lossless compression arise in more complex architectures? The most salient limitation of this work is the setting of single-hidden-layer hyperbolic tangent networks. This setting determines some of the details of our algorithms, theorem statements, and proofs. On the other hand, studying this concrete case has allowed us to analysed three fundamental forms of redundancy that can arise in any neural network, and to outline ways to exploit them for lossless compression. In particular, redundancies arising from zero, constant, or proportional units (cf. reducibility conditions (i)–(iii)) are derived from the computational structure of neuron layers and are present regardless of the choice of nonlinearity.

In this way, our results constitute a meaningful first step towards a theory of lossless compressibility in more modern architectures built from individual feed-forward layers with any nonlinearity. In more complex architectures, these compression opportunities will remain, but there may also be *additional* forms of redundancy that arise, for example due to different affine symmetries of the chosen nonlinearity or due to interactions between layers in a multi-layer network. Such additional forms of redundancy have not been catalogued. The lossless compression framework offers a principled objective to aid in identifying them and responding to them.

We call for future work to ask the question, what does optimal lossless compression look like in more complex architectures? Such work can use our algorithms and analysis for the single-hidden-layer case as a starting point, extending these to exploit novel redundancies in addition to the fundamental redundancies we have identified. In turn, such work can suggest new empirical questions arising from this new perspective on approximate compressibility in learned neural networks.

Acknowledgements

The contributions in this paper also appear in MFR's minor thesis (Farrugia-Roberts, 2022, §4). MFR received financial support from the Melbourne School of Engineering Foundation Scholarship and the Long-Term Future Fund while completing this research. MFR would like to thank Daniel Murfet and Rohan Hitchcock for providing helpful feedback during this research and during the preparation of this manuscript.

References

Francesca Albertini, Eduardo D. Sontag, and Vincent Maillot. Uniqueness of weights for neural networks. In *Artificial Neural Networks for Speech and Vision*, pages 113–125. Chapman & Hall, London, 1993. Proceedings of a workshop held at Rutgers University in 1992. Access via Eduardo D. Sontag. Cited on page 2.

Shun-ichi Amari, Hyeyoung Park, and Tomoko Ozeki. Singularities affect dynamics of learning in neuromanifolds. *Neural Computation*, 18(5):1007–1065, 2006. Access via Crossref. Cited on page 2.

Shun-ichi Amari, Tomoko Ozeki, Ryo Karakida, Yuki Yoshida, and Masato Okada. Dynamics of learning in MLP: Natural gradient and singularity revisited. *Neural Computation*, 30(1):1–33, 2018. Access via Crossref. Cited on page 2.

Caglar Aytekin, Francesco Cricri, and Emre Aksu. Compressibility loss for neural network weights. 2019. Preprint arXiv:1905.01044 [cs.LG]. Cited on page 2.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 535–541. ACM, 2006. Access via Crossref. Cited on page 1.

Stephen Casper, Xavier Boix, Vanessa D'Amario, Ling Guo, Martin Schrimpf, Kasper Vinken, and Gabriel Kreiman. Frivolous units: Wider networks are not really that wide. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence*, volume 8, pages 6921–6929. AAAI Press, 2021. Access via Crossref. Cited on page 6.

An Mei Chen, Haw-minn Lu, and Robert Hecht-Nielsen. On the geometry of feedforward neural network error surfaces. *Neural Computation*, 5(6):910–927, 1993. Access via Crossref. Cited on page 2.

Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. Model compression and acceleration for deep neural networks: The principles, progress, and challenges. *IEEE Signal Processing Magazine*, 35(1):126–136, 2018. Access via Crossref. Cited on pages 2 and 7.

Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. A survey of model compression and acceleration for deep neural networks. 2020. Preprint arXiv:1710.09282v9 [cs.LG]. Updated version of Cheng et al. (2018). Cited on page 2.

Tejalal Choudhary, Vipul Mishra, Anurag Goswami, and Jagannathan Sarangapani. A comprehensive survey on model compression and acceleration. *Artificial Intelligence Review*, 53(7):5113–5155, 2020. Access via Crossref. Cited on page 2.

Florent Cousseau, Tomoko Ozeki, and Shun-ichi Amari. Dynamics of learning in multilayer perceptrons near singularities. *IEEE Transactions on Neural Networks*, 19(8):1313–1328, 2008. Access via Crossref. Cited on page 2.

Matthew Farrugia-Roberts. *Structural Degeneracy in Neural Networks*. Master's thesis, School of Computing and Information Systems, The University of Melbourne, 2022. Access via Matthew Farrugia-Roberts. Cited on page 7.

Matthew Farrugia-Roberts. Functional equivalence and path connectivity of reducible hyperbolic tangent networks. In *Advances in Neural Information Processing Systems 36*, pages 79502–79517. Curran Associates, 2023. Access via NeurIPS. Cited on pages 2, 3, and 10.

Charles Fefferman. Reconstructing a neural net from its output. *Revista Matemática Iberoamericana*, 10(3):507–555, 1994. Access via Crossref. Cited on page 2.

Charles Fefferman and Scott Markel. Recovering a feed-forward net from its output. In *Advances in Neural Information Processing Systems 6*, pages 335–342. Morgan Kaufmann, 1993. Access via NeurIPS. Cited on page 2.

Kenji Fukumizu. A regularity condition of the information matrix of a multilayer perceptron network. *Neural Networks*, 9(5):871–879, 1996. Access via Crossref. Cited on pages 2 and 10.

Kenji Fukumizu and Shun-ichi Amari. Local minima and plateaus in hierarchical structures of multilayer perceptrons. *Neural Networks*, 13(3):317–327, 2000. Access via Crossref. Cited on page 2.

Kenji Fukumizu, Shoichiro Yamaguchi, Yoh-ichi Mototake, and Mirai Tanaka. Semi-flat minima and saddle points by embedding neural networks to overparameterization. In *Advances in Neural Information Processing Systems 32*, pages 13868–13876. Curran Associates, 2019. Access via NeurIPS. Cited on page 2.

Geoffrey E. Hinton and Drew van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, pages 5–13. ACM, 1993. Access via Crossref. Cited on page 2.

Geoffrey E. Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. Presented at Twenty-eighth Conference on Neural Information Processing Systems, Deep Learning workshop, 2014. Preprint arXiv:1503.02531 [stat.ML]. Cited on page 1.

Věra Kůrková and Paul C. Kainen. Functionally equivalent feedforward neural networks. *Neural Computation*, 6(3):543–558, 1994. Access via Crossref. Cited on page 2.

Philipp Petersen, Mones Raslan, and Felix Voigtlaender. Topological properties of the set of functions generated by neural networks of fixed size. *Foundations of Computational Mathematics*, 21(2):375–444, 2021. Access via Crossref. Cited on page 6.

Mary Phuong and Christoph H. Lampert. Functional vs. parametric equivalence of ReLU networks. In *8th International Conference on Learning Representations*. OpenReview, 2020. Access via OpenReview. Cited on page 2.

R. Piziak and P. L. Odell. Full rank factorization of matrices. *Mathematics Magazine*, 72(3):193–201, 1999. Access via Crossref. Cited on page 4.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. Presented at the Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing, 2019. Preprint arXiv:1910.01108 [cs.CL]. Cited on page 1.

Thiago Serra, Abhinav Kumar, and Srikumar Ramalingam. Lossless compression of deep neural networks. In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research: 17th International Conference, CPAIOR 2020, Vienna, Austria, September 21–24, 2020, Proceedings*, pages 417–430. 2020. Access via Crossref. Cited on page 2.

Berfin Şimşek, François Ged, Arthur Jacot, Francesco Spadaro, Clément Hongler, Wulfram Gerstner, and Johanni Brea. Geometry of the loss landscape in overparameterized neural networks: Symmetries and invariances. In *Proceedings of the 38th International Conference on Machine Learning*, pages 9722–9732. PMLR, 2021. Access via PMLR. Cited on page 2.

Héctor J. Sussmann. Uniqueness of the weights for minimal feedforward nets with a given input-output map. *Neural Networks*, 5(4):589–593, 1992. Access via Crossref. Cited on pages 2, 3, 4, and 10.

Taiji Suzuki, Hiroshi Abe, Tomoya Murata, Shingo Horiuchi, Kotaro Ito, Tokuma Wachi, So Hirai, Masatoshi Yukishima, and Tomoaki Nishimura. Spectral pruning: Compressing deep neural networks via spectral analysis and its generalization error. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 2839–2846. IJCAI, 2020a. Access via Crossref. Cited on page 2.

Taiji Suzuki, Hiroshi Abe, and Tomoaki Nishimura. Compression based bound for non-compressed network: Unified generalization error analysis of large compressible deep neural network. In *8th International Conference on Learning Representations*. OpenReview, 2020b. Access via OpenReview. Cited on page 2.

Boris A. Trakhtenbrot. A survey of Russian approaches to *perebor* (brute-force search) algorithms. *Annals of the History of Computing*, 6(4):384–400, 1984. Access via Crossref.

Verner Vlačić and Helmut Bölcskei. Affine symmetries and neural network identifiability. *Advances in Mathematics*, 376:107485, 2021. Access via Crossref. Cited on page 2.

Verner Vlačić and Helmut Bölcskei. Neural network identifiability for a family of sigmoidal nonlinearities. *Constructive Approximation*, 55(1):173–224, 2022. Access via Crossref. Cited on page 2.

Sumio Watanabe. *Algebraic Geometry and Statistical Learning Theory*. Cambridge University Press, 2009. Cited on page 2.

Haikun Wei, Jun Zhang, Florent Cousseau, Tomoko Ozeki, and Shun-ichi Amari. Dynamics of learning near singularities in layered networks. *Neural Computation*, 20(3):813–843, 2008. Access via Crossref. Cited on page 2.

Susan Wei, Daniel Murfet, Mingming Gong, Hui Li, Jesse Gell-Redman, and Thomas Quella. Deep learning is singular, and that's good. *IEEE Transactions on Neural Networks and Learning Systems*, 34(12):10473–10486, 2023. Access via Crossref. Cited on page 2.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations*. OpenReview, 2017. Access via OpenReview. Cited on pages 1 and 9.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021. Access via Crossref. Republication of Zhang et al. (2017). Cited on page 1.

A Hyperbolic tangent networks with multi-dimensional inputs and outputs

In the main paper we consider single-hidden-layer hyperbolic tangent networks with a single input unit and a single output unit. Our algorithms and results generalise to an architecture with multiple input units and multiple output units, with some minor changes.

Consider a family of fully-connected, feed-forward neural network architectures with $n \in \mathbb{N}^+$ input units, $m \in \mathbb{N}^+$ biased linear output units, and a single hidden layer of $h \in \mathbb{N}$ biased hidden units with the hyperbolic tangent nonlinearity. The weights and biases of the network are encoded in a parameter vector in the format $w = (a_1, b_1, c_1, \ldots, a_h, b_h, c_h, d) \in \mathcal{W}_h^{n,m} = \mathbb{R}^{(n+m+1)h+m}$, where for each hidden unit $i = 1, \ldots, h$ there is an *outgoing weight vector* $a_i \in \mathbb{R}^m$, an *incoming weight vector* $b_i \in \mathbb{R}^n$, and a bias $c_i \in \mathbb{R}$; and $d \in \mathbb{R}^m$ is a vector of output unit biases. Thus each parameter $w \in \mathcal{W}_h^{n,m}$ indexes a function $f_w : \mathbb{R}^n \to \mathbb{R}^m$ such that $f_w(x) = d + \sum_{i=1}^h a_i \tanh(b_i \cdot x + c_i)$.

The above notation is deliberately chosen to parallel the case $n = m = 1$ considered in the main paper. This makes the generalisation of our results to the case $n, m \geq 1$ straightforward. First, replace all mentions of the scalar incoming and outgoing weights with incoming and outgoing weight *vectors*. It remains to note the following additional changes.

1. The algorithms and proofs commonly refer to the sign of an incoming weight. For $b \in \mathbb{R}^n$ define $\operatorname{sign}(b) \in \{-1, 0, +1\}$ as the sign of the first nonzero component of b, or zero if $b = 0$. Use this generalised sign function throughout when $n > 1$.
2. Partitioning the (signed) incoming weight and bias pairs of the hidden units is a key part of Algorithms 4.1 and 4.2. An efficient way to compute a partition of a list of values is by first sorting the list. This requires a linear order on the values. When $n = 1$ we can use the lexicographic ordering of pairs of scalars. When $n > 1$ we have pairs of the form $(b, c) \in \mathbb{R}^n \times \mathbb{R}$, and we can lexicographically sort by the tuple $(b_1, \ldots, b_n, c)$.
3. The reducibility conditions were proven by Sussmann (1992) in the case $n \geq 1$ and $m = 1$. The conditions also hold for $m \geq 1$ (see Fukumizu, 1996; or Farrugia-Roberts, 2023, §A).