
Proceedings Paper:
Hankala, T., Hannula, M., Kontinen, J. et al. (1 more author) (2024) Complexity of neural network training and ETR: extensions with effectively continuous functions. In: Proceedings of the 38th AAAI Conference on Artificial Intelligence. 38th Annual AAAI Conference on Artificial Intelligence, 20-27 Feb 2024, Vancouver, Canada. Association for the Advancement of Artificial Intelligence , pp. 12778-12285. ISBN 9781577358879

https://doi.org/10.1609/aaai.v38i11.29118

Complexity of Neural Network Training and ETR: Extensions with Effectively Continuous Functions

Teemu Hankala[1], **Miika Hannula**[1], **Juha Kontinen**[1], **Jonni Virtema**[2]

[1]University of Helsinki, Finland
[2]University of Sheffield, UK
teemu.hankala@helsinki.fi, miika.hannula@helsinki.fi, juha.kontinen@helsinki.fi, j.t.virtema@sheffield.ac.uk

Abstract

The training problem of neural networks (NNs) is known to be $\exists\mathbb{R}$-complete with respect to ReLU and linear activation functions. We show that the training problem for NNs equipped with arbitrary activation functions is polynomial-time bireducible to the existential theory of the reals extended with the corresponding activation functions. For effectively continuous activation functions (e.g., the sigmoid function), we obtain an inclusion to low levels of the arithmetical hierarchy. Consequently, the sigmoid activation function leads to the existential theory of the reals with the exponential function, and hence the decidability of training NNs using the sigmoid activation function is equivalent to the decidability of the existential theory of the reals with the exponential function, a long-standing open problem. In contrast, we obtain that the training problem is undecidable if sinusoidal activation functions are considered.

1 Introduction

Neural networks (NNs) constitute the driving force behind the remarkable achievements of modern artificial intelligence and machine learning. A crucial stage in their development is training, during which NNs adapt to training data and form the capability to generalize their learned patterns to novel scenarios. The objective is to set up the NN's internal parameters in a way that minimizes the selected cost function. In practice, this involves gradually adjusting the NN's weights and biases, making use of cost function gradients at each step. Due to its inherently local nature, and since the cost function is not necessarily convex, the process is not guaranteed to converge to the cost function's global minimum. It is then natural to ask: What is the computational cost of training neural networks in a globally optimized manner? The corresponding decision problem is to decide whether it is even possible to train the NN to work under a given threshold error.

This paper studies the computational complexity of neural network training problems parameterized by various activation functions. Such a training problem asks, given a neural network architecture, finite training data, a cost function and a threshold, whether there exist real-valued edge weights and neuron biases for the network such that the total training error with respect to the training data is below the given threshold value. Neural network training problems have been studied extensively (see (Šíma 2002) for a comprehensive historical survey). Recently, certain versions of this problem were shown to be complete for the complexity class $\exists\mathbb{R}$ (Abrahamsen, Kleist, and Miltzow 2021; Bertschinger et al. 2022), defined as the class of all decision problems P that are polynomial-time reducible to the existential theory of the reals (ETR). This class has turned out to have many interesting complete problems, such as the so-called art gallery problem (Abrahamsen, Adamaszek, and Miltzow 2022), two-dimensional packing problems (Abrahamsen, Miltzow, and Seiferth 2020), and certain decision problems on symmetric Nash equilibria (Bilò and Mavronicolas 2017).

Hardness results for complexity classes above NP give theoretical explanations on why methods suitable for solving problems in NP do not work for the training problem. The activation functions in the $\exists\mathbb{R}$-complete training problems mentioned above are restricted to linear functions and the rectified linear unit (ReLU). In practice, there are also other useful activation functions, such as the standard logistic sigmoid function, which is defined in terms of the exponential function, and sinusoidal activation functions. However, whereas ETR is NP-hard and included in PSPACE (Canny 1988), it is a long-standing and influential open problem posed by Alfred Tarski whether exponential arithmetic is decidable. This problem is known to be related to another major open question – *Schanuel's conjecture* – in transcendental number theory (see, e.g., (Wilkie 1997; Servi 2008)). On the other hand, the theory of the reals extended by the sine function is known to be undecidable, even under some further restrictions on the allowed syntax of the formulae (Richardson 1968).

In this article, we generalize the connection between ETR and the neural network training problem from linear functions and the ReLU activation function to arbitrary real-valued functions. On the ETR side, the corresponding activation function is added as a new function symbol to the signature of the underlying structure $(\mathbb{R}, +, \times, <, =, 0, 1)$. We show that the NN training problem using f as an activation function together with the identity activation function is complete for the extension $\exists\mathbb{R}_f$ of the class $\exists\mathbb{R}$. In other words, this training problem is polynomial-time many-one bireducible to the existential theory of the reals extended with f. Besides a single activation function f, we allow the use of a set τ of activation functions both in the training problem and in the

definition of complexity classes of the form $\exists\mathbb{R}_\tau$. Our results imply that the decidability of the training problem of neural networks using the sigmoid activation function is equivalent to the decidability of (the existential fragment of) exponential arithmetic. In fact, due to a certain model theoretic property (*model completeness*) of the theory of exponential arithmetic, this theory is decidable if and only if its existential fragment is decidable, and for showing decidability, it suffices to show it to be recursively enumerable (Σ_1^0) (see, e.g., (Servi 2008) for further discussion). Another immediate consequence is that the NN training problem is undecidable if the sine function is allowed to be used for neural activation. Interestingly, sinusoidal activation functions have already been observed to be hard to train in simulations, and have been analyzed, for instance, in (Lapedes and Farber 1987). Our result presents a theoretical explanation for the observation.

Moreover, we establish upper bounds for the complexity of the NN training problem in the case of *effectively continuous* activation functions. The class of effectively continuous functions (see (Servi 2008)) contains, for example, the sigmoid function and sinusoidal activation functions. Using rational approximations and basic topological properties, we show that it is possible to place these problems into Σ_3^0, the third level of the arithmetical hierarchy. For effectively continuous functions, the complexity of $\exists\mathbb{R}_\tau$ drops to Σ_1^0 if the use of the equality sign is disallowed in the formulae. Note that disallowing equalities in favour of the strict order relation has no implications in the complexity of ETR without further functions added (Schaefer and Stefankovic 2017).

Related work. The NN training problem has been studied in numerous articles over the past decades, e.g., (Blum and Rivest 1992; Judd 1988; Jones 1997; Goel et al. 2021; Boob, Dey, and Lan 2022). There are only two previous works that relate the existential theory of the reals to the training problem (Abrahamsen, Kleist, and Miltzow 2021; Bertschinger et al. 2022). These articles explore the training problem with respect to piecewise linear activation functions such as ReLU. The problem has not been studied before in conjunction with non-linear activation functions using logical methods. We extend the examination to arbitrary activation functions such as the logistic sigmoid function and sinusoidal activation functions. We settle a question presented in (Abrahamsen, Kleist, and Miltzow 2021) regarding the complexity of the training problem over the sigmoid function: the problem is as hard as the existential theory of the reals with exponentiation. Our logic-based approach is very general as it encompasses all practically relevant activation functions.

2 Preliminaries

We assume familiarity with basic complexity classes such as P, NP and PSPACE, and the arithmetic hierarchy (see, e.g., (Arora and Barak 2009). For real numbers a and b, we let (a, b) and $[a, b]$ denote the open and closed intervals with endpoints a and b, respectively. We write $|a|$ for the absolute value of a.

The first-order language of *real arithmetic*, written $\mathrm{FO}(+, \times, <, =, 0, 1)$, is given by the grammar

$$\phi ::= i < i \mid i = i \mid \phi \wedge \phi \mid \phi \vee \phi \mid \exists x\phi \mid \forall x\phi, \qquad (1)$$

where i stands for numerical terms given by the grammar

$$i ::= 0 \mid 1 \mid x \mid i{\times}i \mid i{+}i, \quad \text{where } x \text{ is a first-order variable.}$$

Note that adding negation to (1) does not increase the expressiveness of the language. A negated formula can be expressed positively via a negation normal form transformation and subsequent positive rewriting of negated atomic formulae.

We consider various extensions of the above language. For a set of relation and function symbols $\mathcal{C}$, $\mathrm{FO}(\mathcal{C})$ is the variant that uses only function and relation symbols in $\mathcal{C}$. *Existential real arithmetic* $\exists\mathrm{FO}(+, \times, <, =, 0, 1)$ is obtained from (1) by dropping universal quantification; the logic $\exists\mathrm{FO}(\mathcal{C})$ is defined analogously.

A formula ϕ is *open* if some variable appears free in ϕ, and otherwise *closed*. Closed formulae are referred to as *sentences*. Given some first-order structure $\mathfrak{A}$ and a variable assignment s, we write $\mathfrak{A} \models_s \phi$ if ϕ is true in $\mathfrak{A}$ with respect to s. If ϕ is a sentence, we write $\mathfrak{A} \models \phi$ if ϕ is true in $\mathfrak{A}$.

The semantics for the language of real arithmetic is defined over the fixed structure $(\mathbb{R}, +, \times, <, =, 0, 1)$ of real arithmetic in the usual way, and similarly for $\mathrm{FO}(\mathcal{C})$ where the symbols in $\mathcal{C}$ have their usual interpretations. As a slight abuse of notation, we use f to denote both a real-valued function and the corresponding function symbol. Moreover, we allow function symbols to be interpreted as partial functions. Any atomic formula with some undefined term with respect to an assignment s is defined to be false.

We write $\mathbb{R}$ and $\mathbb{R}^<$ as shorthands for the structures $(\mathbb{R}, +, \times, <, =, 0, 1)$ and $(\mathbb{R}, +, \times, <, 0, 1)$, respectively. For a collection τ of additional functions, we write $\mathbb{R}_\tau$ and $\mathbb{R}_\tau^<$ for the corresponding extension of the models $\mathbb{R}$ and $\mathbb{R}^<$, respectively. We define the complexity class $\exists\mathbb{R}_\tau$ as the collection of decision problems that have a polynomial-time many-one reduction to closed formulae of the existential real arithmetic with additional functions from τ. We also consider a subclass of $\exists\mathbb{R}_\tau$ defined in terms of strict inequalities: the class $\exists\mathbb{R}_\tau^<$ is defined otherwise as $\exists\mathbb{R}_\tau$, except that we drop equality atoms $i = i$ from (1).

3 Effectively Continuous Functions and the Arithmetical Hierarchy

We begin by showing that $\exists\mathbb{R}_\tau$ and $\exists\mathbb{R}_\tau^<$ are included in low levels of the arithmetical hierarchy, when the functions in τ are *effectively continuous*.

Definition 1 ((Servi 2008, Definition 4.2.1))**.** *A function $f\colon \mathbb{R}^n \to \mathbb{R}$ is* effectively continuous *if there exists a computable function g with the following properties:*

- *To each n-tuple of open intervals $(a_1, b_1), \ldots, (a_n, b_n)$ with rational endpoints, the function g associates an open interval (c, d) with rational endpoints such that*

$$\forall x \left(x \in (a_1, b_1) \times \cdots \times (a_n, b_n) \implies f(x) \in (c, d)\right).$$

- *$\forall M > 0\ \forall \varepsilon > 0\ \exists \delta > 0\ \forall a_i, b_i$ it holds that, if c, d is the output of g on input a_i, b_i, then*

$$\bigwedge_{i\in[n]} (a_i, b_i) \subseteq [-M, M] \wedge |a_i - b_i| < \delta \implies |c - d| < \varepsilon.$$

In this case, it is said that g computes *the function f. We write* ECF *to denote the set of all effectively continuous functions.*

As noted in (Servi 2008), there is at most one effectively continuous function for each computable function. Thus the set ECF is countable, and not every constant function is effectively continuous. On the other hand, rational constant functions, the identity function $x \mapsto x$, the absolute value function $x \mapsto |x|$, together with addition, multiplication, division, and the exponential function and basic trigonometric functions are effectively continuous. Using these functions and the following lemma, all rational functions and many standard neural activation functions, such as the ReLU function $x \mapsto \max(0, x)$ and the sigmoid function $\sigma\colon x \mapsto 1/(1 + \exp(-x))$, can be seen to be in ECF.

Lemma 2 ((Servi 2008, Lemma 4.2.6))**.** ECF *is effectively closed under composition. That is, given computable functions g and g' which compute two effectively continuous functions f and f', respectively, we can effectively find a computable function g'' which computes the composition $f \circ f'$ (when the latter is defined).*

Lemma 3 ((Servi 2008, Remark 4.2.2 (rephrased)))**.** *Effectively continuous functions are uniformly continuous on every compact cube of the form $[-M, M]^n$, where $M > 0$. In particular, effectively continuous functions are continuous in the Euclidean topology.*

The proof of the following theorem can be found in the full arXiv version (Hankala et al. 2023). The result relies on the fact that, in the absence of the equality sign, existential formulae over $\exists\mathbb{R}^{<}_{\mathrm{ECF}}$ define open sets in the standard Euclidean topology.

Theorem 4. *Let $\phi(\bar{x})$ be an existential formula without identity over the structure $\mathbb{R}_{\mathrm{ECF}}$. Then the set of rational solutions $\bar{a} \in \mathbb{Q}^n$ for $\mathbb{R}_{\mathrm{ECF}} \models \phi(\bar{a})$ is recursively enumerable. Moreover, the complexity class $\exists\mathbb{R}^{<}_{\mathrm{ECF}}$ is contained in Σ^0_1.*

In the following, a *τ-polynomial* is a τ-term where, except for the arithmetic operations $+$ and $\times$, functions g can only appear with variable or constant arguments (i.e., in the form $g(x)$ or $g(c)$). Furthermore, the (total) degree of a polynomial P is defined as the maximum over the sum of the variable exponents in each monomial term occurring in P. Note that the minus sign can be avoided by switching monomial terms from one side of the (in)equations to the other.

Lemma 5. *Let τ be a set of function symbols, and let $\phi(\bar{x}) \in \exists\mathrm{FO}(\{+, \times, <, =, 0, 1\} \cup \tau)$. Then we can construct, in polynomial time with respect to the length of $\phi(\bar{x})$, a τ-polynomial $P(\bar{x}, \bar{y})$ of degree at most 4 such that $\phi(\bar{x})$ is equivalent to $\exists\bar{y} P(\bar{x}, \bar{y}) = 0$ over the expanded reals $\mathbb{R}_\tau$.*

Proof. The proof is analogous to that of (Schaefer and Stefankovic 2017, Lemma 3.2). In fact, it suffices to first simplify all subterms of ϕ of the form $g(t)$ to the required form using new variables: replace $g(t)$ by $g(y)$ and add a new conjunct $y = t$. Then the translation in (Schaefer and Stefankovic 2017) can be used directly by treating terms $g(y)$ exactly in the same way as in the original proof. □

Analogous to the $\exists\mathbb{R}$-complete decision problem 4-FEAS of (Schaefer and Stefankovic 2017), we define 4-FEAS$_\tau$ to be the decision problem that asks, given a τ-polynomial of degree at most 4, if the polynomial is feasible, i.e., has a root in the extended model $\mathbb{R}_\tau$. Using the previous lemma, we obtain a complete problem for the analogue of $\exists\mathbb{R}$ with functions in τ, thus justifying the use of the notation $\exists\mathbb{R}_\tau$.

Corollary 6. 4-FEAS$_\tau$ *is $\exists\mathbb{R}_\tau$-complete.*

Using Lemma 5, we can now give an upper bound for the complexity class of the existential theory of the reals extended with effectively continuous functions.

Theorem 7. $\exists\mathbb{R}_{\mathrm{ECF}}$ *is contained in Σ^0_3.*

Proof. Let ϕ be an existential sentence over $\mathbb{R}_{\mathrm{ECF}}$. By Lemma 5, we may assume that ϕ is of the form $\exists\bar{x} P_\tau(\bar{x}) = 0$, where P_τ is a τ-polynomial over a vocabulary τ that contains function symbols for the effectively continuous functions. By Lemmas 2 and 3, the function defined by P_τ is effectively continuous. We show that $\exists\bar{x} P_\tau(\bar{x}) = 0$ is equivalent with

$$\exists d > 0\ \forall \varepsilon > 0\ \exists \bar{x} \in B(\bar{0}, d) : |P_\tau(\bar{x})| < \varepsilon \qquad (2)$$

over $\mathbb{R}_{\mathrm{ECF}}$. Clearly $\exists\bar{x} P_\tau(\bar{x}) = 0$ implies (2). For the converse direction, (2) entails that there is an infinite sequence of tuples $(\bar{x}_i)_{i\in\mathbb{N}}$ inside some open ball $B(\bar{0}, d)$ such that $\lim_{i\to\infty} P_\tau(\bar{x}_i) = 0$. Since the closure $\bar{B}(\bar{0}, d)$ is compact, the sequence $(\bar{x}_i)_{i\in\mathbb{N}}$ has a subsequence $(\bar{y}_j)_{j\in\mathbb{N}}$ that converges to a limit point $y \in \bar{B}(\bar{0}, d)$. Since P_τ is continuous, and $\lim_{j\to\infty} P_\tau(\bar{y}_j) = 0$, we obtain that $P_\tau(\bar{y}) = 0$ and, in particular, that the claim $\exists\bar{x} P_\tau(\bar{x}) = 0$ holds.

The parameters ε and d can without loss of generality be assumed to be rational in (2). The innermost existentially quantified part of (2) then describes a predicate that is definable by an existential formula with ε and d as rational constant parameters over $\mathbb{R}^{<}_{\mathrm{ECF}}$ and thus is in Σ^0_1 by Theorem 4. The full expression (2) hence defines a Σ^0_3 predicate. Since $\exists\mathbb{R}_{\mathrm{ECF}}$ is defined as the class of problems that can be reduced in polynomial time to sentences of the form of ϕ, we conclude that $\exists\mathbb{R}_{\mathrm{ECF}}$ is contained in Σ^0_3. □

The following results follow from the preceding theorems.

Corollary 8. $\exists\mathbb{R}^{<}_{\exp} \subseteq \Sigma^0_1$ *and* $\exists\mathbb{R}_{\exp} \subseteq \Sigma^0_3$.

In fact, by the model completeness of the first-order theory of $\mathbb{R}_{\exp}$ (see (Wilkie 1997; Servi 2008)), the decidability of its existential fragment is equivalent to it being recursively enumerable, and also equivalent to the full first-order theory of $\mathbb{R}_{\exp}$ being decidable. Thus, the precise relationship of $\exists\mathbb{R}_{\exp}$ to Σ^0_1 and Σ^0_3 is an open problem. These observations can be generalized to any $\mathbb{R}_\tau$ with a model-complete first-order theory and to any set τ of effectively continuous functions.

Below, for convenience of stating the theorem, we assume that $\exists\mathbb{R}_{\sin}$ and $\exists\mathbb{R}^{<}_{\sin}$ are closed under computable reductions.

Theorem 9. $\Sigma^0_1 \subseteq \exists\mathbb{R}_{\sin} \subseteq \Sigma^0_3$ *and* $\exists\mathbb{R}^{<}_{\sin} = \Sigma^0_1$.

Proof. Note that the formula $\sin(y) = 0 \wedge 4 < y \wedge y < 7$ defines 2π in y, and $\sin(2\pi \times x) = 0 \wedge (x = 0 \vee x > 0)$ defines natural numbers over the reals, with the help of the previously defined 2π. The first Σ^0_1-hardness follows from the Davis–Putnam–Robinson–Matiyasevich theorem. The latter hardness follows from the results of (Laczkovich 2003). □

4 The Neural Network Training Problem

We show that, under a suitable formulation, the neural network training problem is complete for $\exists\mathbb{R}_\tau$, where τ consists of the unary real functions that are used for neural activation. In the case of effectively continuous activation functions, the complexity of the training problem is then in Σ_3^0.

Definition 10. *A* neural network architecture *is a pair $N = (G, \varphi)$ over a finite directed acyclic graph $G = (V, E)$ and a function φ that maps each $v \in V$ to some function $\varphi(v)\colon \mathbb{R} \to \mathbb{R}$. The vertices of the graph are called* neurons. *An* input (resp. output) neuron *is a neuron that has no incoming (resp. outgoing) edges. Every other vertex of the graph is a* hidden neuron. *For each neuron v, the value $\varphi(v)$ is the* activation function *of the neuron v. The function $\varphi(v)$ is also denoted as φ_v.*

Definition 11. *A* neural network *is a triple $\mathcal{N} = (N, w, b)$ over some neural network architecture N together with real functions w and b with the following properties:*

- *The function w maps each edge e of the underlying graph to a real number. The value $w(e)$ is the* edge weight *of e in the network and is alternatively denoted as w_e.*
- *The function b maps each non-input neuron v to a real number. This is called the* bias *of v and is denoted as b_v.*

A *data point* for an architecture N is a function d that maps a real number to each input and output neuron.

Definition 12. *Let $\mathcal{N} = (N, w, b)$ be some neural network. By the* neural function *of $\mathcal{N}$ we mean the unique function g with the following definitions and properties:*

- *The domain of g is the set of all possible data points for the architecture N. For each data point d, the value $g(d)$ is a function that maps each neuron v to a real number, called the* neural value *of v computed by the network for the data point d. This function is also referred to as g_d.*
- *For each data point $d \in \mathrm{dom}(g)$ and each input neuron v it holds that $g_d(v) = d(v)$.*
- *For each $x \in \mathrm{dom}(g)$ and for each non-input neuron $v \in V$ it holds that*

$$g_d(v) = \varphi_v\left(b_v + \sum_{u \in P}\big(w_{(u,v)} \times g_d(u)\big)\right), \qquad (3)$$

where P is the set of all immediate predecessors of v in the underlying directed acyclic graph.

In other words, the neural function for a data point d is defined recursively as follows: For each input neuron v, the value given by the neural function is the value $d(v)$ given by the data point. For every other neuron v, the neural value $g_d(v)$ is computed by first adding the bias value b_v to the sum of the incoming neural values, each multiplied by the corresponding edge weight, and then mapping the resulting sum through the activation function φ_v. The neural function is well defined, since the network architecture is assumed to be acyclic.

In the following definition, we denote by d_O the tuple that consists of the values of a data point d for the output neurons of the network architecture, where some fixed order on the output neurons independent of the data points is assumed.

Definition 13. *Let τ be some fixed set of unary real-valued functions. An* NN_τ-TRAINING instance *is a tuple $(N, A_E, A_V, D, c, \prec, \delta)$ as follows:*

- *The pair $N = (G, \varphi)$ is a neural network architecture.*
- *For each neuron v, the activation function φ_v is either the identity function $x \mapsto x$ or a function f from τ.*
- *A subset A_E of the edges of the graph G mark the* active edges *in the training process. Similarly, A_V is the subset of* active neurons.
- *D is a finite set of data points that map non-hidden neurons to rational numbers.*
- *A* cost function $c\colon \mathbb{R}^{2m} \to \mathbb{R}$, *where m is the number of output neurons of N, is assumed to be definable with an arithmetic expression.*
- *The symbol $\prec$ is one the of relations in the set $\{=, \leq, <\}$.*
- *A* threshold *value δ for the allowed total training error is given as a rational number.*

A pair (w, b) is a satisfying solution *for the training instance $(N, A_E, A_V, D, c, \prec, \delta)$, if (N, w, b) is a neural network (as in Definition 11) such that $w(e) = 1$ for each edge $e \in E \setminus A_E$, $b(v) = 0$ for each neuron $v \in V \setminus A_V$, and the neural function g satisfies the condition*

$$\sum_{d \in D} c\left((g_d)_O, d_O\right) \prec \delta. \qquad (4)$$

NN_τ-TRAINING *is the decision problem that asks, given an* NN_τ-TRAINING *instance, whether it has a satisfying solution.* NN_τ-TRAINING$^<$ *is defined likewise, except that the strict order relation $<$ is the only allowed choice for $\prec$. For a function f, we write* NN_f-TRAINING *and* NN_f-TRAINING$^<$ *for* $\mathrm{NN}_{\{\mathrm{f}\}}$-TRAINING *and* $\mathrm{NN}_{\{\mathrm{f}\}}$-TRAINING$^<$, *respectively.*

A cost function c is *faithful*, if $c(\bar{a}, \bar{b}) = 0$ if and only if $\bar{a} = \bar{b}$, for all $\bar{a}$ and $\bar{b}$. If the cost function c is both faithful and non-negative, the threshold value δ equals 0, and $\prec$ is replaced with the equality sign, then the condition 4 is equivalent to all data points $d \in D$ satisfying $(g_d)_O = d_O$. Then $g_d(v)$ equals $d(v)$ for every output neuron of the network.

Lemma 14. *For any set τ of unary real-valued functions,* NN_τ-TRAINING$^<$ *is in the complexity class $\exists\mathbb{R}^<_{\tau \cup \{-, \div\}}$.*

Proof. Let $(N, A_E, A_V, D, c, \prec, \delta)$ be an instance for the problem NN_τ-TRAINING$^<$. We show how to construct a formula $\phi(\bar{w}, \bar{b})$ of $\exists\mathrm{FO}(\{+, -, \times, \div, <, 0, 1\} \cup \tau)$ in polynomial time so that there exists a satisfying solution for the given instance of the training problem if and only if $\mathbb{R}_\tau \models \exists\bar{w}\,\exists\bar{b}\,\phi(\bar{w}, \bar{b})$, where $\bar{w}$ and $\bar{b}$ are finite sequences of variables that in some fixed order encode all the weights and biases of the active edges and active neurons, respectively. We use g to denote the intended neural function of N that corresponds to the variables in $\bar{w}$ and $\bar{b}$. Let m be the number of output neurons of N.

First we show that for each data point $d \in D$ and for each neuron v of the network, there is a term $t_{(d,v)}$ which evaluates to $g_d(v)$ over $\mathbb{R}_\tau$. Namely, if v is an input neuron, we can express $t_{(d,v)}$ succinctly as the corresponding input value given by d, for instance, in the form of a fraction of

two binary expansions of integral values. If v is a non-input neuron, the term $t_{(d,v)}$ can be obtained as in the expression 3 of Definition 12 using the terms $t_{(d,u)}$ of the predecessors u of v and the sets A_E and A_V. Note that the underlying graph structure is given as a part of the input, and that the graph is assumed to be acyclic.

As in Definition 13, the cost function c is expressible as an arithmetic expression. Then, for each $d \in D$, there is some term t'_d that evaluates to $c((g_d)_O, d_O)$ over $\mathbb{R}_\tau$. Furthermore, the given rational threshold value δ can be expressed by some constant term t''_δ. Finally, the relational formula 4 of Definition 13 is expressible as $\sum_{d \in D} t'_d < t''_\delta$, thus yielding a formula $\phi(\bar{w}, \bar{b})$ as claimed. □

Note that in the previous proof, $\phi(\bar{w}, \bar{b})$ is even a single quantifier-free relational atom. It also defines the set of all satisfying solutions for the training instance as a subset of the Euclidean space $\mathbb{R}^{|A_E|+|A_V|}$. If all the functions in τ are effectively continuous, this solution set is open in the Euclidean topology and its intersection with $\mathbb{Q}^{|A_E|+|A_V|}$ is recursively enumerable by Theorem 4. In particular, in this case the training instance has a solution if and only if it has a rational solution.

If the first-order theory of the model $\mathbb{R}_\tau$ is *o-minimal* (see, e.g., (Servi 2008) for definition and discussion), then the set of satisfying solutions has only a finite number of connected components. In particular, the structure $\mathbb{R}_{\exp}$ is known to be o-minimal, whereas $\mathbb{R}_{\sin}$ is not o-minimal. However, for example in the case of the sigmoid activation function, it is an open problem whether positive lower bounds for the radii of open neighbourhoods for solutions in the open solution set can be computed in general. Considering the model completeness of $\mathbb{R}_{\exp}$, this question is in part connected to Tarski's exponential function problem and the *first root conjecture* (see, e.g., (Wilkie 1997; Servi 2008)).

Lemma 15. *For any set τ of unary real-valued functions, the decision problem* NN_τ-TRAINING *is in* $\exists\mathbb{R}_\tau$.

Proof. Let $(N, A_E, A_V, D, c, \prec, \delta)$ be an instance for the problem NN_τ-TRAINING and let the formula $\phi(\bar{w}, \bar{b})$ be as in the proof of Lemma 14. It is enough to show that there is some formula $\phi'(\bar{w}, \bar{b})$ of $\exists\mathrm{FO}(\{+, \times, <, =, 0, 1\} \cup \tau)$ that is equivalent to $\phi(\bar{w}, \bar{b})$ over the structure $\mathbb{R}_{\tau \cup \{-, \div\}}$. For this, note that if $\theta(x)$ is a formula with some open variable x, then formulae of the forms $\theta(-y)$ and $\theta(y/z)$ can be written as $\exists x(\theta(x) \wedge x + y = 0)$ and $\exists x(\theta(x) \wedge x \times z = y)$. Thus, the functions $-$ and $\div$ can be eliminated from $\phi(\bar{w}, \bar{b})$. □

Notice that the proof of Lemma 15 and the membership of the training problem in the class $\exists\mathbb{R}_\tau$ can be generalized to the case where all data points, activation functions, threshold values and cost functions can be defined using existential formulae of the alphabet $\{+, \times, <, =, 0, 1\} \cup \tau$. In addition, the set τ may also include other than unary functions.

We define a decision problem for existential formulae in a certain normal form. The definition is an extended version of the problem called ETR-INV in (Abrahamsen, Adamaszek, and Miltzow 2022), now allowing unary function symbols.

Definition 16. *Let τ be a set of unary real-valued functions. Then* ETR_τ-INV-FLAT *is the following decision problem: Given a finite set $\mathcal{C}$ of* constraints*, each of one of the forms*

$$x = 1, \;\; x + y + z = 0, \;\; x \times y + 1 = 0, \;\; x + f(y) = 0, \quad (5)$$

where x, y and z are first-order variable symbols and f is some function in τ, determine whether the constraints of $\mathcal{C}$ are satisfiable over $\mathbb{R}_\tau$ using a single variable assignment.

The different constraint types listed in 5 are called *unit* constraints, *addition* constraints, *inversion* constraints and *function* constraints, respectively.

Lemma 17. ETR_τ-INV-FLAT *is $\exists\mathbb{R}_\tau$-complete.*

Proof. It is enough to show that the satisfiability problem of the existential theory of $\mathbb{R}_\tau$ is reducible in polynomial time to ETR_τ-INV-FLAT. Let ϕ be a sentence of the logic $\exists\mathrm{FO}(\{+, \times, <, =, 0, 1\} \cup \tau)$. First simplifying all subterms that correspond to function symbols as in the proof of Lemma 5 and then proceeding in the same manner as in the proof of Theorem 16 of (Abrahamsen, Adamaszek, and Miltzow 2022), the satisfiability of ϕ can be reduced in polynomial time to a finite set $\mathcal{C}'$ of constraints of the following forms:

$$x = 1, \quad x + y = z, \quad x \times y = 1, \quad x = f(y).$$

For every variable x, y and z, the condition $x + y + z = 0$ is under every variable assignment equivalent to

$$\exists u \exists v(v + v + v = 0 \wedge z + u + v = 0 \wedge x + y + u = 0).$$

Similarly, for monomial terms t and t', the formula $t = t'$ is under any variable assignment equivalent to

$$\exists u \exists v(v + v + v = 0 \wedge t + u + v = 0 \wedge t' + u + v = 0).$$

Thus, each of the constraints in $\mathcal{C}'$ can be expressed in an equivalent manner using at most a constant number of ETR_τ-INV-FLAT constraints. □

Depending on the functions in τ, some of the constraints in Definition 16 can be removed without losing completeness for $\exists\mathbb{R}_\tau$. For instance, the exponential function can be used in order to switch between addition and multiplication in existential formulae.

The proof of the following theorem is based on the neural network construction of (Abrahamsen, Kleist, and Miltzow 2021), in which it was used to obtain the first known version of an $\exists\mathbb{R}$-complete NN training problem.

Theorem 18. *Let τ be a set of unary real-valued functions. Then* NN_τ-TRAINING *is $\exists\mathbb{R}_\tau$-complete.*

Proof. By Lemmas 15 and 17 it is enough to show that the decision problem ETR_τ-INV-FLAT has a polynomial-time reduction to NN_τ-TRAINING. To this end, let $\mathcal{C}$ be some finite set of ETR_τ-INV-FLAT constraints. We may assume that $\mathcal{C}$ does not include unsatisfiable constraints of the form $x \times x + 1 = 0$. Similarly to (Abrahamsen, Kleist, and Miltzow 2021), we may further assume that each variable symbol is incident with at most one inversion constraint. Let W be the set of variable symbols that appear in the constraints of $\mathcal{C}$.

We describe how to construct in polynomial time such an NN_τ-TRAINING instance $(N, A_E, A_V, D, c, \prec, \delta)$ that

has a satisfying solution if and only if the finite system $\mathcal{C}$ of constraints is satisfiable. First, let $(A_V, \prec, \delta) := (\varnothing, =, 0)$. In particular, biases are assumed to be 0 for all neurons. Let c be any faithful and non-negative cost function. We may refer to the variables of individual constraints using indices in the set $\{1, 2, 3\}$ according to the subscripts in the following order for each constraint type:

$$x_1 = 1, \qquad x_1 + x_2 + x_3 = 0,$$
$$x_1 \times x_2 + 1 = 0, \qquad x_1 + f(x_2) = 0. \tag{6}$$

For each constraint $C \in \mathcal{C}$, let $l_C \in \{1, 2, 3\}$ be the total number of these indices for C. We define the underlying graph G of the network architecture $N = (G, \varphi)$ based on the sets W and $\mathcal{C}$. The resulting substructure of the network for each constraint type is depicted in Figure 1.

- For each variable $x \in W$, there is a unique input neuron i_x and an edge to its immediate successor j_x. The intended meaning of these two neurons is to encode the value of the variable x into the weight of the edge (i_x, j_x).
- There is a unique output neuron o_C for each constraint C of $\mathcal{C}$. For each index $k \in \{1, \ldots, l_C\}$ there are unique hidden neurons $h_{(C,k)}$ and $q_{(C,k)}$, and an input neuron $p_{(C,k)}$. They are incident with the edges $(j_{x_k}, h_{(C,k)})$, $(h_{(C,k)}, o_C)$, $(p_{(C,k)}, q_{(C,k)})$ and $(q_{(C,k)}, h_{(C,k)})$. Here, the neurons $p_{(C,k)}$ and $q_{(C,k)}$ are intended to cancel out the output value of the neuron $h_{(C,k)}$ for those data points that in the construction are not intended to directly concern the constraint C but that still give a non-zero value for the input neuron i_{x_k}.
- For each constraint of the form $x \times y + 1 = 0$, an input neuron e_C is added and connected to the neuron j_y using the edge (e_C, j_y).

Finally, the function φ is selected so that for each neuron v, the activation function φ_v is the identity function except for the case that v is of type $h_{(C,2)}$ for some function constraint C of the form $x_1 + f(x_2) = 0$; in this case, the selection $\varphi_v := f$ is used. The set A_E of active edges is defined to be the set of all edges of N that are either of the form (i_x, j_x), $(j_y, h_{(C,2)})$ or $(p_{(C',k)}, q_{(C',k)})$, where x and y are variables, C is an inversion constraint $z \times y + 1 = 0$ for some $z \in W$, and $C' \in \mathcal{C}$ is any constraint for which $k \in \{1, \ldots, l_{C'}\}$ is an appropriate index. In Figure 1, active edges are drawn using thicker arrows than the other edges of the network.

Recall that all input neurons of N are either of the form i_x, $p_{(C,k)}$ or e_C, and that $\{o_C \mid C \in \mathcal{C}\}$ is the set of all output neurons. Let D consist of the following three data points:

- The data point $d_a \in D$ is defined as follows:
  - For each variable $x \in W$ we have $d_a(i_x) = 1$. For each constraint C of the form $x \times y + 1 = 0$ it holds that $d_a(p_{(C,1)}) = 1$ and $d_a(p_{(C,2)}) = 1$. For every other input neuron v of the network, $d_a(v) = 0$.
  - For every output neuron o_C corresponding to some unit constraint $C \in \mathcal{C}$, it holds that $d_a(o_C) = 1$. For every other output neuron v, we have $d_a(v) = 0$.
- The remaining two data points, d_l and d_r, are associated with inversion constraints and are defined as follows:

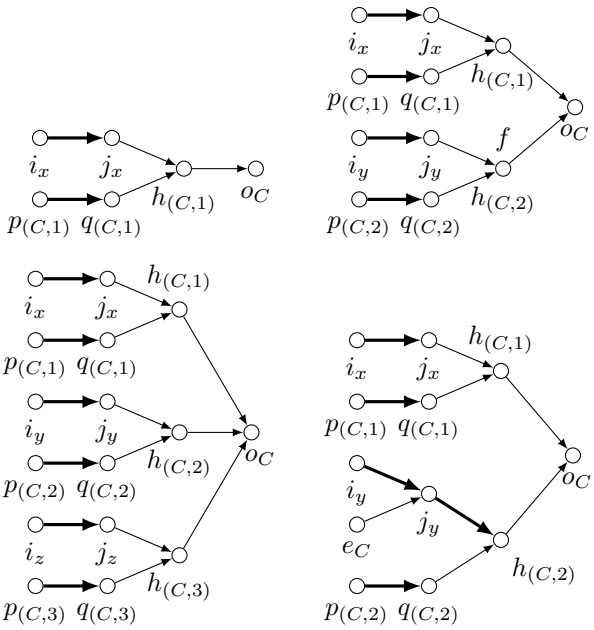

Figure 1: Left to right, top to bottom: The graph structures, or gagdets, corresponding to unit constraints, function constraints, addition constraints and inversion constraints.

  - If C is an inversion constraint of the form $x \times y + 1 = 0$, it holds that the data point d_l maps the neuron tuple (i_x, i_y, e_C, o_C) to the tuple $(1, 0, 1, 0)$, whereas the data point d_r maps the tuple (i_x, i_y, e_C, o_C) to $(0, 1, 0, 1)$.
  - If $C' \in \mathcal{C}$ is not an inversion constraint, d_l maps to 1 all those input neurons of the form $p_{(C',k)}$ for which the number $k \in \{1, \ldots, l_{C'}\}$ is an index for x_1 in some inversion constraint $x_1 \times x_2 + 1 = 0$, and d_r maps to 1 all the input neurons of the form $p_{(C',k)}$ for which $k \in \{1, 2, 3\}$ is an index for the variable x_2 in some inversion constraint $x_1 \times x_2 + 1 = 0$ of $\mathcal{C}$.
  - If v is any other input neuron or output neuron of the network N, it holds that $d_l(v) = 0$ and $d_r(v) = 0$.

Note that in the preceding definition of the data points d_l and d_r we use the assumption that $\mathcal{C}$ does not have constraints of the form $x \times x + 1 = 0$ and that each variable appears in at most one inversion constraint.

It remains to show that the NN_τ-TRAINING instance $(N, A_E, A_V, D, c, \prec, \delta)$ has a satisfying solution if and only if the system $\mathcal{C}$ is satisfiable. For this, first assume that some edge weight function w is a solution for the training problem. Let variable assignment s be such that for each variable $x \in W$ it holds that $s(x) = w(i_x, j_x)$. We show that s satisfies the system $\mathcal{C}$. For this, let $C \in \mathcal{C}$ be arbitrary.

- If C is a unit constraint $x = 1$, then by the definition of d_a and the non-negativeness of the cost function c, it holds that $d_a(o_C) = 1$. Since $d_a(i_x) = 1$ and $d_a(p_{(C,1)}) = 0$, we have $w(i_x, j_x) = 1$ and so $s(x) = 1$.
- In the case that C is $x + y + z = 0$, we have $d_a(o_C) = 0$ and $d_a(p_{(C,k)}) = 0$ for each index $k \in \{1, 2, , 3\}$. Thus,

$w(i_x, j_x) + w(i_y, j_y) + w(i_z, j_z) = 0$ or, equivalently, $s(x) + s(y) + s(z) = 0$.

- If C is an inversion constraint of the form $x \times y + 1 = 0$, the definition of d_l and the structure of N imply that

$$w(i_x, j_x) + w(j_y, h_{(C,2)}) = d_l(o_C) = 0.$$

Similarly, from the definition of d_r we get

$$w(j_y, h_{(C,2)}) \times w(i_y, j_y) = d_r(o_C) = 1.$$

Thus both of the claims $w(i_x, j_x) \times w(i_y, j_y) + 1 = 0$ and $s(x) \times s(y) + 1 = 0$ are true.

- If C is a function constraint of the form $x + f(y) = 0$, we have $d_a(o_C) = 1$, and $d_a(i_x) = d_a(i_y) = 1$, and $d_a(p_{(C,k)}) = 0$ for $k \in \{1, 2\}$. From these it follows that $w(i_x, j_x) + f(w(i_y, j_y)) = 0$ and $s(x) + f(s(y)) = 0$.

Therefore, the assignment s satisfies all the constraints of $\mathcal{C}$.

For the other direction, let s be some assignment over the variables in W such that all the constraints of $\mathcal{C}$ are satisfied. Let w be an edge weight function defined as follows:

- For each variable $x \in W$, we have $w(i_x, j_x) = s(x)$.
- For each inversion constraint C of the form $x \times y + 1 = 0$, value $-s(x)$ is assigned to the edge $(j_y, h_{(C,2)})$. The edge $(p_{(C,2)}, q_{(C,2)})$ is mapped to the value $s(x) \times s(y)$.
- Every remaining edge in the set A_E that is of the form $(p_{(C,k)}, q_{(C,k)})$ for some $C \in \mathcal{C}$ is given the value $-s(x_k)$, where x_k is such a variable symbol that it matches with the index $k \in \{1, 2, 3\}$ of C in the sense of the list 6.
- For every $e \in E \setminus A_E$ it holds that $w(e) = 1$.

Let b be the bias function that maps each non-input neuron of N to 0, and let g be the neural function of the neural network (N, w, b). It remains to show that for every output neuron v and for every data point $d \in D$ it holds that $g_d(v) = d(v)$. Recall that for each output neuron v there is a unique $C \in \mathcal{C}$ such that $v = o_C$.

We will first consider the data point d_a. If $C \in \mathcal{C}$ is not an inversion constraint, then for all indices $k \in \{1, \ldots, l_C\}$ it holds that $g_{d_a}(h_{(C,k)})$ evaluates to

$$w(i_{x_k}, j_{x_k}) \times d_a(i_{x_k}) + w(p_{(C,k)}, q_{(C,k)}) \times d_a(p_{(C,k)}),$$

which equals to $s(x_k)$. From this it follows that if C is a unit constraint, then $g_{d_a}(o_C) = s(x_1) = 1$, and if C is an addition constraint, then $g_{d_a}(o_C) = s(x_1) + s(x_2) + s(x_3) = 0$. Furthermore, if C is some function constraint of the form $x_1 + f(x_2) = 0$, then $g_{d_a}(o_C) = s(x_1) + f(x_2) = 0$. Then, for each of the preceding cases we have $g_{d_a}(o_C) = d_a(o_C)$. If C is an inversion constraint $x \times y + 1 = 0$, then the value of $g_{d_a}(h_{(C,1)})$ is obtained by

$$\begin{aligned} &w(i_x, j_x) \times d_a(i_x) + w(p_{(C,1)}, q_{(C,1)}) \times d_a(p_{(C,1)}) \\ &\quad = s(x) \times 1 + (-s(x)) \times 1 = 0, \end{aligned}$$

and in a similar manner, $g_{d_a}(h_{(C,2)})$ can be evaluated to

$$(-s(x) \times (s(y) \times 1 + 0) + (s(x) \times s(y)) \times 1) = 0.$$

Thus, it holds that $g_{d_a}(o_C) = 0 = d_a(o_C)$.

Next, let d be either of the data points d_l and d_r. Then for each constraint $C' \in \mathcal{C}$ that is not an inversion constraint and for each index $k \in \{1, \ldots, l_{C'}\}$ it holds that $h_{(C',k)} = 0$. Namely, either it holds that both of the claims $d(x_k) = 0$ and $d(p_{(C',k)}) = 0$ are true, or that both $d(x_k) = 1$ and $d(p_{(C',k)}) = 1$ are true. Then we have $g(o_{C'}) = 0 = d(o_{C'})$.

For any inversion constraint C of the form $x \times y + 1 = 0$ and for each $d \in \{d_l, d_r\}$, the value $g_d(h_{(C,1)})$ evaluates to

$$w(i_x, j_x) \times d(i_x) + w(p_{(C,1)}, q_{(C,1)}) \times d(p_{(C,1)})$$

and thus is equal to $s(x) \times d(i_x)$. From this it follows that $g_{d_l}(h_{(C,1)}) = s(x)$ and $g_{d_r}(h_{(C,1)}) = 0$. On the other hand, the value $g_d(h_{(C,2)})$ can be evaluated to

$$-s(x) \times (s(y) \times d(i_y) + d(e_C)).$$

Then, $g_{d_l}(o_C) = 0$ and $g_{d_r}(o_C) = 1$, from which it follows that $g_d(o_C) = d(o_C)$ for each $d \in \{d_l, d_r\}$. This concludes the proof the theorem. □

Using the previous theorem, we can connect sigmoidal NN training with exponential real arithmetic.

Corollary 19. NN_σ-TRAINING *is* $\exists\mathbb{R}_{\exp}$*-complete.*

Proof. By Theorem 18, NN_σ-TRAINING is $\exists\mathbb{R}_\sigma$-complete. Furthermore, the sigmoid function σ is existentially definable in $\mathbb{R}_{\exp}$ and similarly the exponential function is existentially definable in $\mathbb{R}_\sigma$. Thus it follows that NN_σ-TRAINING is $\exists\mathbb{R}_{\exp}$-complete. □

Corollary 20. $\mathrm{NN}_{\sin}$-TRAINING *is undecidable but in* Σ^0_3.

Proof. The claim follows from Theorems 9 and 18. □

Similar to the proof of Theorem 9, the NN training problem can be shown to be undecidable in the case of activation functions having some form of periodic nature that can be used to define natural numbers with existential formulae.

5 Conclusion

We studied the complexity of the neural network training problem and showed that the training problem corresponding to sets τ of activation functions are complete for the complexity classes $\exists\mathbb{R}_\tau$ that extend the reals with the corresponding functions. We also showed that for sets τ of effectively continuous functions, these problems reside in Σ^0_3, and even in Σ^0_1 when defined without the equality sign and negation. The following topics deserve further study:

Can we improve or extend our Σ^0_1 upper bound for the identity-free existential theory of the reals with effectively continuous functions? Note that for decidability of the exponential arithmetic it would suffice to establish an Σ^0_1-upper bound for its existential fragment (Wilkie 1997; Servi 2008).

It is an open question whether our result holds if the NN architecture is assumed to be a fully connected graph with all edge weights and neuron biases being active, or when restricted to the case that all neurons use the same activation function. Cf. (Bertschinger et al. 2022).

It is an open problem when the complexities of the general training problem NN_τ-TRAINING and its restrictions NN_τ-TRAINING$^{\leq}$ and NN_τ-TRAINING$^{<}$ coincide. The present paper establishes the upper bounds Σ^0_3 and Σ^0_1 for these restricted versions, respectively.

Acknowledgments

Miika Hannula has been supported by the ERC grant 101020762. Teemu Hankala and Juha Kontinen were partially funded by the Academy of Finland grant 345634. Jonni Virtema was partially supported by the DFG grant VI 1045-1/1 and by the Academy of Finland grant 345634.

References

Abrahamsen, M.; Adamaszek, A.; and Miltzow, T. 2022. The Art Gallery Problem is $\exists\mathbb{R}$-complete. *J. ACM*, 69(1): 4:1–4:70.

Abrahamsen, M.; Kleist, L.; and Miltzow, T. 2021. Training Neural Networks is ER-complete. In *NeurIPS*, 18293–18306.

Abrahamsen, M.; Miltzow, T.; and Seiferth, N. 2020. Framework for ER-Completeness of Two-Dimensional Packing Problems. In *FOCS*, 1014–1021. IEEE.

Arora, S.; and Barak, B. 2009. *Computational Complexity - A Modern Approach*. Cambridge University Press.

Bertschinger, D.; Hertrich, C.; Jungeblut, P.; Miltzow, T.; and Weber, S. 2022. Training Fully Connected Neural Networks is $\exists$R-Complete. *CoRR*, abs/2204.01368.

Bilò, V.; and Mavronicolas, M. 2017. Existential-R-Complete Decision Problems about Symmetric Nash Equilibria in Symmetric Multi-Player Games. In *STACS*, volume 66 of *LIPIcs*, 13:1–13:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Blum, A. L.; and Rivest, R. L. 1992. Training a 3-node neural network is NP-complete. *Neural Networks*, 5(1): 117–127.

Boob, D.; Dey, S. S.; and Lan, G. 2022. Complexity of Training ReLU Neural Network. *Discret. Optim.*, 44(P1).

Canny, J. F. 1988. Some Algebraic and Geometric Computations in PSPACE. In Simon, J., ed., *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, 460–467. ACM.

Goel, S.; Klivans, A.; Manurangsi, P.; and Reichman, D. 2021. Tight Hardness Results for Training Depth-2 ReLU Networks. In Lee, J. R., ed., *12th Innovations in Theoretical Computer Science Conference (ITCS 2021)*, volume 185 of *Leibniz International Proceedings in Informatics (LIPIcs)*, 22:1–22:14. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik. ISBN 978-3-95977-177-1.

Hankala, T.; Hannula, M.; Kontinen, J.; and Virtema, J. 2023. Complexity of Neural Network Training and ETR: Extensions with Effectively Continuous Functions. *CoRR*, abs/2305.11833.

Jones, L. 1997. The computational intractability of training sigmoidal neural networks. *IEEE Transactions on Information Theory*, 43(1): 167–173.

Judd, S. 1988. On the complexity of loading shallow neural networks. *Journal of Complexity*, 4(3): 177–192.

Laczkovich, M. 2003. The Removal of π from Some Undecidable Problems Involving Elementary Functions. *Proceedings of the American Mathematical Society*, 131(7): 2235–2240.

Lapedes, A.; and Farber, R. 1987. Nonlinear signal processing using neural networks: Prediction and system modelling. In *IEEE international conference on neural networks, San Diego, CA, USA, 21 Jun 1987*.

Richardson, D. 1968. Some Undecidable Problems Involving Elementary Functions of a Real Variable. *J. Symb. Log.*, 33(4): 514–520.

Schaefer, M.; and Stefankovic, D. 2017. Fixed Points, Nash Equilibria, and the Existential Theory of the Reals. *Theory Comput. Syst.*, 60(2): 172–193.

Servi, T. 2008. On the First-Order Theory of Real Exponentiation. volume 6 of *Publications of the Scuola Normale Superiore*. Edizioni della Normale Pisa. ISBN 978-88-7642-325-3.

Wilkie, A. J. 1997. *Schanuel's Conjecture and the Decidability of the Real Exponential Field*, 223–230. Dordrecht: Springer Netherlands. ISBN 978-94-015-8923-9.

Šíma, J. 2002. Training a Single Sigmoidal Neuron Is Hard. *Neural Computation*, 14(11): 2709–2728.