Complexity of Reachability Problems in Neural Networks

Adrian Wurm[(✉)]

BTU Cottbus-Senftenberg, Lehrstuhl Theoretische Informatik, Platz der Deutschen
Einheit 1, 03046 Cottbus, Germany
`wurm@b-tu.de`
`https://www.b-tu.de/`

Abstract. In this paper we investigate formal verification problems for
Neural Network computations. Various reachability problems will be in the
focus, such as: Given symbolic specifications of allowed inputs and outputs
in form of Linear Programming instances, one question is whether valid
inputs exist such that the given network computes a valid output? Does
this property hold for all valid inputs? The former question's complexity
has been investigated recently in [20] by Sälzer and Lange for nets using
the Rectified Linear Unit and the identity function as their activation func-
tions. We complement their achievements by showing that the problem is
NP-complete for piecewise linear functions with rational coefficients that
are not linear, NP-hard for almost all suitable activation functions includ-
ing non-linear ones that are continuous on an interval, complete for the
Existential Theory of the Reals $\exists\mathbb{R}$ for every non-linear polynomial and
$\exists\mathbb{R}$-hard for the exponential function and various sigmoidal functions. For
the completeness results, linking the verification tasks with the theory of
Constraint Satisfaction Problems turns out helpful.

1 Introduction

Given the huge success of utilizing Neural Networks, NN for short, in the last decade,
such nets are nowadays widely used in all kind of data processing, including tasks
of varying difficulty. There is a wide range of applications, the following exemplary
references (mostly taken from [20]) just collect non-exclusively some areas for fur-
ther reading: Image recognition [15], natural language processing [9], autonomous
driving [8], applications in medicine [16], and prediction of stock markets [7], just
to mention a few. Khan et al. [14] provide a survey of such applications, a math-
ematically oriented textbook concerning structural issues related to Deep Neural
Networks is provided by [4]. Among the many different aspects of areas where the
use of Neural Networks seems appropriate, some also involve safety-critical systems
like autonomous driving or power grid management. In such a setting, when security
issues become important, aspects of certification come into play [10].

In the present paper we are interested in studying certain verification prob-
lems for NNs in form of particular reachability problems. Starting point is the
work by Sälzer and Lange [20] being based on [13,18]. The authors of these

papers analyze the computational complexity of one particular such verification task. It deals with a reachability problem in networks using the Rectified Linear Unit together with the identity function as its activation function. In such a net, specifications describe the set of valid inputs and outputs in form of two Linear Programming instances. The question then is to decide whether a valid input exists such that the network's result on that input is a valid output, i.e., whether the set of valid outputs is reachable from that of valid inputs. In the above references the problem is shown to be NP-complete, even for one hidden layer and output dimension one, or some restricted set of weights being used [20]. Note that the network in principle is allowed to compute with real numbers, so the valid inputs we are looking for belong to some space $\mathbb{R}^n$, but the network itself is specified by its discrete structure and rational weights defining the linear combinations computed by its single neurons.

Obviously, one can consider a huge variety of networks created by changing the underlying activation functions. There are of course many activations frequently used in NN frameworks, and in addition we could extend reachability questions to nets using all kinds of activation. One issue to be discussed is the computational model in which one argues. If, for example, the typical sigmoid activation $f(x) = 1/(1 + e^{-x})$ is used, it has to be specified in which sense it is computed by the net: For example exactly or approximately, and at which costs these operations are being performed.

In the present work we study the reachability problem for commonly used activation functions and show that for most of them it will be complete either in (classical) NP or in the presumably larger class $\exists\mathbb{R}$, which captures the so-called existential theory of the reals. Our main results are as follows: The reachability problem is in P for linear activations, in NP for semilinear activations, NP-hard for all non-linear activations that are continuous on an interval, and ETR-hard for several commonly used activations such as arctan and the exponential function. These results imply, for example, NP-completeness for frequently used activations such as (Leaky) ReLU, Heaviside and Signum.

A most helpful tool for establishing these results is linking the problems under consideration to the area of constraint satisfaction problem CSP and known complexity results for special instances of the latter. This connection will provide us with a classification of a vast set of activation functions in the complexity classes between P and $\exists\mathbb{R}$. We also consider a variant of the reachability problem asking whether for all valid inputs the computed output is necessarily valid and establish several complexity results as well.

The paper is organized as follows: In Sect. 2 we collect basic notions, recall the definition of feedforward neural nets as used in this paper as well as useful facts about Constraint Satisfaction Problems. Section 3 studies various activation functions and their impact on the complexity of reachability problems. We show that the reachability problem is basically the same as the CSP containing the graphs of the activation functions together with relations necessary to express linear programming instances. We show that adding the identity as activation does not change the complexity of the reachability problem in several cases, for example when either ReLU is used as activation or if a network connection is

allowed to skip a layer. We show that the problem is NP-hard for every sensible non-linear activation and finally discuss problems that are hard or complete for $\exists\mathbb{R}$. The paper ends with some open questions. Lacking proofs are given in the full version.

2   Preliminaries and Network Reachability Problems

We start by defining the problems we are interested in; here, we follow the definitions and notions of [20] for everything related to neural networks. The networks considered are exclusively feedforward. In their most general form, they can process real numbers and contain rational weights.

Definition 1. *A (feedforward) neural network N is a layered graph that represents a function of format $\mathbb{R}^n \to \mathbb{R}^m$, for some $n, m \in \mathbb{N}$. The first layer with label $\ell = 0$ is called the* input layer *and consists of n nodes called* input nodes. *The input value x_i of the i-th node is also taken as its output $y_{0i} := x_i$. A layer $1 \le \ell \le L - 2$ is called* hidden *and consists of $k(\ell)$ nodes called* computation nodes. *The i-th node of layer ℓ computes the output $y_{\ell i} = \sigma_{\ell i}(\sum_j c_{ji}^{(\ell-1)} y_{(\ell-1)j} + b_{\ell i})$. Here, the $\sigma_{\ell i}$ are (typically nonlinear)* activation *functions (to be specified later on) and the sum runs over all output neurons of the previous layer. The $c_{ji}^{(\ell-1)}$ are real constants which are called* weights, *and $b_{\ell i}$ is a real constant called* bias. *The outputs of all nodes of layer ℓ combined gives the output $(y_{\ell 0}, ..., y_{\ell(k-1)})$ of the hidden layer. The final layer $L - 1$ is called* output layer *and consists of m nodes called* output nodes. *The i-th node computes an output $y_{(L-1)i}$ in the same way as a node in a hidden layer. The output $(y_{(L-1)0}, ..., y_{(L-1)(m-1)})$ of the output layer is considered the output $N(x)$ of the network N.*

Note that above, as in [20], we allow several different activation functions in a single network. This basically is because for some results technically the identity is necessary as a second activation function beside the 'main' activation function used. We next recall from [20] the definition of the reachability problem NNREACH. Since we want to study its complexity in the Turing model, we restrict all weights and biases in a NN to the rational numbers. The problem involves two Linear Programming LP instances in a decision version, recall that such an instance consists of a system of (componentwise) linear inequalities $A \cdot x \le b$ for a rational matrix A and vector b of suitable dimensions. The decision problem asks for the existence of a real solution vector x.

Definition 2. *a) Let F be a set of activation functions from $\mathbb{R}$ to $\mathbb{R}$. An instance of the* reachability problem for neural networks NNREACH(F) *consists of an $n \in \mathbb{N}$, a (feedforward) neural network N with n inputs and all its activation functions belonging to F, rational data as weights and biases, and two instances of LP in decision version with rational data, one with the input variables of N as variables, and the other with the output variables of N as variables. These*

instances are also called input *and* output specification, *respectively. The problem is to decide if there exists an $x \in \mathbb{R}^n$ that satisfies the input specification such that the output $N(x)$ satisfies the output specification.*

b) The problem verification of interval property VIP *(F) consists of the same instances, except for the output specification being the open polyhedron, meaning the interior of the solution space. This is due to technical reasons that will later on simplify the reductions. The question is whether for all $x \in \mathbb{R}^n$ satisfying the input specification, $N(x)$ will satisfy the output specification (cf. [10]).*

As for NNReach*, we denote by (A, B, N) such an instance, assuming n is obvious from the context.*

c) Let $F = \{f_1, ..., f_n\}$ be a set of activation functions. Then the Network Equivalence problem NE*(F) is the decision problem whether two F-networks describe the same function or not.*

d) The size of an instance is given as the sum of the (usual) bit-sizes of the two LP instances and $T \cdot L$; here, T denotes the number of neurons in the net N and L is the maximal bit-size of any of the weights and biases.

As usual for neural networks, we consider different choices for the activation functions used. Typical activation functions are $ReLU(x) = max\{0, x\}$, the Heaviside function or sigmoidal functions like $\sigma(x) = \frac{1}{1+e^{-x}}$. By technical reasons, in some situations the identity function $\sigma(x) = x$ is also allowed, Sälzer and Lange [20], for example, examined NNReach$(id, ReLU)$. We name nodes according to their internal activation function, so we call nodes with activation function $\sigma(x) = x$ identity nodes and nodes with activation function $\sigma(x) = ReLU(x)$ ReLU-nodes etc. Note that the terminology of the LP-specifications has its origin in software verification.

2.1   Basics on Constraint Satisfaction Problems CSP

As we shall see, analyzing the complexity of the above reachability problems is closely related to suitable questions in the framework of Constraint Satisfaction Problems CSP. This is a well established area in complexity, see for example the survey [6]. Here, we collect the basic notions and results necessary for our purposes.

Informally, a CSP deals with the question whether values from a set A can be assigned to a set of variables so that given conditions (constraints) hold. These conditions are taken from a set of relations over A that, together with the set A, define the CSP. This can be formalized as follows:

Definition 3. *A (relational)* signature *is a pair $\tau = (\mathbf{R}, a)$, where $\mathbf{R}$ is a finite set of* relation symbols *and $a \colon \mathbf{R} \to \mathbb{N}$ is a function called the* arity.

A (relational) τ-structure is a tuple $\mathcal{A} = (A, \mathbf{R}^{\mathcal{A}})$, where A is a set called the domain *and $\mathbf{R}^{\mathcal{A}}$ is a set containing precisely one relation $R^{\mathcal{A}} \subseteq A^{a(R)}$ for each relation symbol $R \in \mathbf{R}$.*

An instance of a CSP over a given τ-structure is a conjunction of constraints, where a single constraint restricts a variable tuple to belong to a particular

relation of the structure under a suitable assignment of values from the domain to the variables. For the entire instance one then asks for the existence of an assignment satisfying all its constraints.

Definition 4. *Let τ be a signature and $\mathcal{A}$ a τ-structure with domain A and relations $\mathbf{R}$. We always assume equality to be among the structure's relations. Let $X = \{x_1, x_2, \ldots\}$ be a countable set of variables.*

a) A constraint *for $\mathcal{A}$ is an expression $R(y_1, ..., y_{a(R)})$, where $R \in \mathbf{R}, a(R)$ its arity and all $y_i \in X$. For $z \in A^{a(R)}$ we say that $R(z)$ is* true over $\mathcal{A}$ *iff $z \in R^{\mathcal{A}}$.*

b) A formula ψ is called primitive positive *if it is of the form*

$$\exists x_{n+1}, ..., \exists x_t \colon \psi_1 \wedge ... \wedge \psi_k,$$

where each ψ_i either is a constraint, $\top$ (true), or $\bot$ (false).

A formula with no free variables is a sentence.

c) The decision problem $\mathrm{CSP}(\mathcal{A})$ *is the following: Let $n \in \mathbb{N}$ and a primitive positive τ-sentence over X, i.e., a finite collection of constraints involving variables $\{x_1, \ldots, x_n\} \subset X$ be given. The question then is, whether there exists an assignment $f : \{x_1, \ldots, x_n\} \to A$ for the variables such that each given constraint is true under the assignment f, i.e., all $R(f(y_1), ..., f(y_{a(R)}))$ are true.*

The size *of an instance is $n + m$, where m denotes the number of constraints.*

Example 1 (folklore). Consider as domain the real numbers $\mathbb{R}$, together with the binary order relation $\leq$, the ternary relation R_+ defined via $R_+(x, y, z) \Leftrightarrow x + y = z$, and the unary relation $R_{=1}(x) \Leftrightarrow x = 1$. Then $\mathrm{CSP}(\mathbb{R}; \leq, R_+, R_{=1})$ is polynomial time equivalent to the Linear Programming problem in feasibility form with rational input data. Reducing the former to the latter is obvious, for the reverse direction first multiply all inequalities with a sufficiently large natural number to obtain integer coefficients only. Now observe that any natural number n can be expressed as (one component of) a solution of a set of constraints involving $a = 1$ and doubling a number via $c = b + b$. This way, the binary expansion of n can be constructed with $O(\log n)$ constraints. Apply this construction similarly to a variable x of an instance of LP to obtain the term $n \cdot x$; now adding as constraint the equation $nx = 1$ similarly allows to express rational numbers as coefficients. Clearly the size of the resulting CSP instance is polynomially bounded in the (bit-)size of the given LP instance. Note that due to the theory of Linear Programming an instance with rational data has the same answer, independently of whether the considered domain is $\mathbb{R}$ or $\mathbb{Q}$.

Definition 5. *A relation R is called* primitive positive definable *(pp-definable) over $\mathcal{A}$, iff it can be defined by a primitive positive formula ψ over $\mathcal{A}$, i.e.,*

$$R(x_1, ..., x_n) \Leftrightarrow \exists x_{n+1}, ..., \exists x_t \colon \psi(x_1, \ldots, x_t).$$

It was shown by Jeavons, Bulatov and Krokhin [3] that $\mathrm{CSP}(\mathcal{A})$ and $\mathrm{CSP}(\mathcal{A}')$, where the latter structure arises from the former by attaching finitely many relations being pp-definable over $\mathcal{A}$, are linear-time equivalent. The obvious idea of replacing every occurrence of the new relation suffices to prove the statement. This argument will be used below once in a while.

Definition 6. *a) A set $R \subseteq \mathbb{R}^n$ is called* semilinear, *iff it is a boolean combination of half-spaces*[1]*.*

b) A set $R \subseteq \mathbb{R}^n$ is called essentially convex, *iff for any two points $x, y \in \mathbb{R}^n$ the intersection of the line segment $[x, y]$ contains only finitely many points that are not in R. If $R \subseteq \mathbb{R}^n$ is not essentially convex, any two points for which the property fails are called* witnesses *for the set not being essentially convex.*

This gives us access to the following results of Bodirsky, Jonsson and von Oertzen:

Theorem 1 ([2]). *a) Let $R_1, ..., R_n$ be semilinear relations. Then $\mathrm{CSP}(\mathbb{Q}; \leq, R_+, R_{=1}, R_1, ..., R_n)$ is in P if $R_1, ..., R_n$ are essentially convex and is NP-complete otherwise.*

b) Let $R_1, ..., R_n$ be relations such that at least one of them is not essentially convex witnessed by two rational points. Then $\mathrm{CSP}(\mathbb{R}; \leq, R_+, R_{=1}, R_1, ..., R_n)$ is NP-hard.[2]

3   Complexity Results for Reachability

We shall now study the complexity of the reachability problem for various sets of activation functions used by the neural network under consideration. Starting point will be the result from [13,20] that $\mathrm{NNReach}(id, ReLU)$ is NP-complete. We analyze the problem for a larger repertoire of activation functions. To do so, in a first step it will be very helpful to relate these problems to instances of certain CSP problems which can be attached to a network canonically. This relation is made precise in the following theorem. The fact that input and output specifications are LP instances causes, that the structures below naturally contain the relations $R_{=1}, R_+$, and $\leq$. Further relations then will be determined by the activation functions used.

Theorem 2. *For any set of unary real functions $F = \{f_1, ..., f_s\}$, interpreted as relations via their graphs, $\mathrm{CSP}(\mathbb{R}; \leq, R_+, R_{=1}, f_1, ..., f_s)$ and $\mathrm{NNReach}(id, f_1, ..., f_s)$ are linear-time equivalent.*

Proof. We prove both directions explicitly for the case $s = 1$, then the conclusion for $s > 1$ is immediate. For reducing $\mathrm{NNReach}(id, f)$ to $\mathrm{CSP}(\mathbb{R}; \leq, R_+, R_{=1}, f)$, let N be a network using id and f as activation functions. The weights and biases of N are assumed to be rational numbers. The variable set of the CSP we construct contains one variable for each input and output node of N. For each node v in a hidden layer we introduce two variables v_{sum} and v_f. Note that according to Example 1 any linear inequality with rational coefficients can be expressed as an instance of $\mathrm{CSP}(\mathbb{R}; \leq, R_+, R_{=1})$ of linear size. Thus, the

---

[1]   i.e., finite unions, intersections and complements of sets of the form $Ax \leq b$.

[2]   Note that we can not switch between the domains $\mathbb{Q}$ and $\mathbb{R}$ at will any more after dropping semilinearity with rational coefficients, for it could in this case change solvability.

input and output specifications of N can be expressed as a set of constraints in $\text{CSP}(\mathbb{R}; \leq, R_+, R_{=1})$ using the corresponding variables, which is of linear size with respect to the size of those specifications. For the nodes in the hidden layers, we proceed similarly. If node v receives a linear sum $\sum_{i=1}^{k} c_i \cdot u_i + b$ as its input, where c_i, b are the rational weights and bias and the u_i represent the outputs of the previous layer, then as in Example 1 we add the constraint $v_{sum} = \sum_{i=1}^{k} c_i \cdot u_{i,f} + b$ to the constructed instance. In case v has f as activation, we add the constraint $v_f = f(v_{sum})$ and if v was an id-node we add the constraint $v_f = v_{sum}$. Obviously, the size of the CSP instance is linearly bounded in that of the given net. Moreover, $\text{NNREACH}(id, f)$ is solvable for N if and only if the above CSP has a solution.

For the reverse direction, we translate an instance of $\text{CSP}(\mathbb{R}; \leq, R_+, R_{=1}, f)$ into an instance of $\text{NNREACH}(id, f)$ with only one hidden layer. For each variable in the instance we introduce a node in the input layer and encode all constraints of the form $\leq, R_+$ and $R_{=1}$ into the input specification. For every constraint $y = f(x)$ we introduce a new f-node $\bar{x}$ in the hidden layer connected only to x with bias 0 and weight 1. Next, we allocate an identity-node $\bar{y}$ in the hidden layer connected only to y also with bias 0 and weight 1 for the connections. Finally, we require both nodes $\bar{x}$ and $\bar{y}$ to be equal by adding the equation $\bar{x} = \bar{y}$ to the output-specification.

It is obvious that both reductions can be performed inductively for all the functions in F, so the statement holds for the entire set. ∎

Note that the proof does not depend on formalizing the specifications as LP instances. It would similarly hold if the specifications would be given by (in-)equality systems involving polynomials and adding a relation for multiplication on the CSP-side. However, in this case checking feasibility of the specifications is already difficult, see below.

Before studying NNREACH for different activations, we briefly discuss a more technical issue, namely the necessity of adding id as activation.

The above proof implies that using an injective activation allows to omit id, if we drop the condition that the network has to be layered, meaning that a connection can skip layers:

Lemma 1. *For f injective, $\text{NNREACH}(id, f)$ and $\text{NNREACH}(f)$ are linear-time equivalent.*

Proof. Given the proof of Theorem 2 it only remains to avoid id-nodes when reducing an instance of $\text{CSP}(\mathbb{R}; \leq, R_+, R_{=1}, f)$ to one of $\text{NNREACH}(f)$. Identity nodes were used to propagate the value of a node y in order to include a constraint $y = f(x)$ in the output specification. Instead, if f is injective one can use a network node for $f(x)$ and one for y and connect them with biases 0 and weights 1 and -1, respectively, to an f-activation node computing $f(f(x) - y)$. Use another f-node to compute $f(0)$. This is possible by demanding a further input node to have value 0. Finally, in the output specification we add the

equality between these two nodes with values $f(f(x) - y)$ and $f(0)$; injectivity provides the equivalence of this condition with $f(x) = y$. ∎

For example, for nets using sigmoidal activation functions identity activations are not necessary.

Sälzer and Lange [20] asked whether for the NP-completeness result involving id and $ReLU$ as activations one can avoid id as activation. Though $ReLU$ is not injective, this in fact holds as well, even when we do not allow connections to skip layers:

Proposition 1. *The following problems are linear-time equivalent:*

i) $\mathrm{CSP}(\mathbb{R}; \leq, R_+, R_{=1}, ReLU)$,
ii) $\mathrm{NNREACH}(id, ReLU)$ *and*
iii) $\mathrm{NNREACH}(ReLU)$ *with only one hidden layer.*

As consequence, all three are NP-complete.

Proof. Given the NP-completeness of $\mathrm{NNREACH}(id, ReLU)$ and Theorem 2 above, it remains to reduce $\mathrm{NNREACH}(id, ReLU)$ to $\mathrm{NNREACH}(ReLU)$ in linear time. Towards this goal, we show that an identity node can be replaced by two ReLU-nodes in the following way: In the neural net to be constructed use two copies of the identity node and let both have the same incoming and outgoing connections as the original node. Replace the identity map by the ReLU map in both and invert all incoming and outgoing weights as well as the bias in the second one. Delete the initial identity node. This does not change the computed function of the network, because

$$\sum_{i=1}^{n} a_i x_i + b = max\{0, \sum_{i=1}^{n} a_i x_i + b\} + min\{0, \sum_{i=1}^{n} a_i x_i + b\}$$
$$= max\{0, \sum_{i=1}^{n} a_i x_i + b\} - max\{0, -(\sum_{i=1}^{n} a_i x_i + b)\}$$
$$= ReLU(\sum_{i=1}^{n} a_i x_i + b) - ReLU(-(\sum_{i=1}^{n} a_i x_i + b))$$
$$= ReLU(\sum_{i=1}^{n} a_i x_i + b) - ReLU(\sum_{i=1}^{n}(-a_i)x_i - b)$$

Applying this to every node gives us at most twice as many nodes with at most four times as many connections, thus the construction runs in linear time. ∎

Theorem 2 enables us treating network reachability complexity questions by using the rich fund of complexity results for CSP problems of various types.

As an easy warm up, convince yourself that $\mathrm{NNREACH}(id)$ is by the previous theorem equivalent to $\mathrm{CSP}(\mathbb{R}; \leq, R_+, R_{=1}, id)$ which in turn is equivalent to LP by Example 1, a problem well known to belong to P in Turing model complexity [12].

In [20] it was shown that $\mathrm{NNReach}(id, ReLU)$ is NP-complete, the link to CSPs however provides us with a much shorter proof. We will make use of Theorem 1 and apply Theorem 2:

Corollary 1. *Let $f_1, ..., f_s$ be unary real functions. If their graphs $g_1, ..., g_s$ are semilinear with rational coefficients, then $\mathrm{NNReach}(id, f_1, ..., f_s)$ is in P if and only if $g_1, ..., g_s$, interpreted as binary relations, are essentially convex, and NP-complete otherwise. If at least one of the graphs $g_1, ..., g_s$ is not essentially convex witnessed by two rational points, then $\mathrm{NNReach}(id, f_1, ..., f_s)$ is NP-hard.*

Though Corollary 1 certainly is not that surprising from the CSP point of view, it gives an elegant way to argue about the complexity of reachability problems for neural networks. Of interest now is to investigate the latter for many more activation functions. NP-hardness is for example the case for $f(x) = n^x, n \in \mathbb{N}, n > 1$, the witnesses are $f(0) = 1$ and $f(1) = n$, but not necessarily for $f(x) = e^x$ for it lacks a second rational point. Other activation functions that immediately give us NP-hardness this way are all non-linear rational functions (including polynomials) with rational coefficients, the square-root, all other rational roots, and the binary logarithm. Reasonable non-linear functions that lack rational points can be compressed to do so, such as $f(x) = sin(\frac{x}{\pi})$ instead of $f(x) = sin(x)$.

Moreover, if we use the strong version that requires semilinearity, we get that $\mathrm{NNReach}(id, f)$ is NP-complete for f being either the absolute value, the floor or ceiling function on a bounded domain, the sign or the Heaviside function, piecewise linear functions such as $f_\alpha(x) = \begin{cases} -1 & if\ x \leq -\alpha \\ \frac{x}{\alpha} & if\ -\alpha < x < \alpha \\ 1 & if\ x \geq \alpha \end{cases}$, the ReLU function, Leaky ReLU given via $R_\alpha(x) = \begin{cases} x & if\ x \geq 0 \\ \alpha x & if\ x < 0 \end{cases}, \alpha \in \mathbb{Q}$ as well as all their scalar generalizations and any other non-trivial rational step function such as the indicator function on an interval.

We will now see a much more powerful version of the NP-hardness part of Corollary 1:

Theorem 3. $\mathrm{NNReach}(id, f)$ *is NP-hard for any non-linear function $f : \mathbb{R} \to \mathbb{R}$ that is continuous on an interval $[a, b] \subseteq \mathbb{R}$, including $f(x) = \frac{x}{1+e^{-x}}$ and all sigmoidal functions such as $f(x) = \frac{1}{1+e^{-x}}$, $f(x) = \frac{x}{\sqrt{1+x^2}}$ and $f(x) = tanh(x)$ the hyperbolic tangent.*

Proof. By the previous Corollary, it suffices to show that with such a function f we can pp-define a new function $\bar{f}$ that excludes an interval witnessed by two rational points. The construction of $\bar{f}$ proceeds in several steps.

First, by density of $\mathbb{Q}$ there exist $\bar{a}, \bar{b} \in [a, b] \cap \mathbb{Q}, \bar{a} < \bar{b}$ so that $f|_{[\bar{a}, \bar{b}]}$ is still non-linear and continuous, we apply a linear transformation on the argument of f so that we can assume $\bar{a} = 0$ and $\bar{b} = 1$. This is pp-definable for rational $\bar{a}, \bar{b}$ and neither changes non-linearity nor continuity. There must exist $c, d \in [0, 1] \cap \mathbb{Q}$, such that

$$f\left(\frac{c+d}{2}\right) \neq \frac{f(c) + f(d)}{2},$$

for f would otherwise fulfill Cauchy's functional equation and therefore be linear.

Apply again a linear transformation on the argument that maps c to 0 and d to 1 and call the resulting function $\hat{f}$. By construction,

$$\hat{f}\left(\frac{1}{2}\right) \neq \frac{\hat{f}(0) + \hat{f}(1)}{2}$$

Define the function $\bar{f} : \mathbb{R} \to \mathbb{R}$ by $\bar{f}(x) = \hat{f}(x) + \hat{f}(1-x) - \hat{f}(0) - \hat{f}(1)$, this is primitive positive, for addition, affine transformation and the constants 0 and 1 are pp-definable in $(\mathbb{R}; \leq, R_+, R_{=1})$. It also matches the requirements for Corollary 1 part 2: The rational points are $\bar{f}(0) = 0$ and $\bar{f}(1) = 0$ and the function must exclude an interval because

$$\bar{f}\left(\frac{1}{2}\right) = \hat{f}\left(\frac{1}{2}\right) + \hat{f}\left(\frac{1}{2}\right) - \hat{f}(0) - \hat{f}(1) = 2(\hat{f}\left(\frac{1}{2}\right) - \frac{\hat{f}(0) + \hat{f}(1)}{2}) \neq 0$$

and by continuity.   ∎

Note that any sensible/common activation function is either linear or of this type. We have shown that in the latter case, together with the identity as activation function, we have an NP-hard reachability problem. Theorem 3 however only states NP-hardness for this vast set of reachability problems. One might wonder about membership in NP. Our next main result will show that membership in NP and thus NP-completeness is unlikely for many of the activations, because reachability becomes complete for a complexity class conjectured to be much larger than NP.

Definition 7 (cf. [19]). *The problem of deciding whether a system of polynomial equations with integer coefficients is solvable over $\mathbb{R}$ is called the* Existential Theory of the Reals *ETR. The complexity class $\exists\mathbb{R}$ is defined as the set of all decision problems that reduce to ETR in polynomial time. A problem is called $\exists\mathbb{R}$-complete if it is in $\exists\mathbb{R}$ and ETR reduces to the problem in polynomial time.*

It was shown by Canny [5] that ETR is in PSPACE and it is easily seen that ETR is NP-hard. These are currently the best known bounds and it is widely believed that $\text{NP} \subsetneq \exists\mathbb{R} \subsetneq \text{PSPACE}$.

ETR can be formulated as $\text{CSP}(\mathbb{R}, E)$, where E is the set of all polynomial relations $R_p := \{x \in \mathbb{R}^n \mid p(x)\nabla 0, \nabla \in \{=, <, \leq\}, p \in \mathbb{Z}[x_1, ..., x_n]\}$ and inequalities. Note that E does not have finite signature any more, this issue has to be resolved before talking about algorithms and complexities of the CSP, for the encoding into Turing Machines is only possible for finite sets of relations. However, E is a first-order reduct of $(\mathbb{R}; 1, +, \cdot)$, any integer polynomial can be described by the integers, addition, multiplication and logical combinations of these like we did for LP in Example 1. It thus suffices to represent the relations by their first-order definition. The integer coefficients can be assumed to be encoded in binary by the same ongoing that we used in Example 1.

The following Theorem will provide us with NNReach problems that are hard or even complete for $\exists\mathbb{R}$.

Theorem 4. *a)* NNREACH(id, f) *is* $\exists\mathbb{R}$*-complete for f any non-linear polynomial and* $\exists\mathbb{R}$*-hard for f any function that coincides with a non-linear polynomial on an interval.*
b) NNREACH(id, f) *is* $\exists\mathbb{R}$*-hard in the following cases:*

i) for any function f that allows to pp-define a function that coincides with the exponential function on an interval, especially the exponential function itself,
$ELU(x) := \begin{cases} x & if \ x \geq 0 \\ \lambda(e^x - 1) & if \ x \leq 0 \end{cases}$ *where $\lambda \in \mathbb{Q}$, and $f(x) = e^{-|x|}$*

ii) for the Gaussian function $f(x) = e^{-x^2}$
iii) for the arctan function $f(x) = arctan(x)$ and

iv) for the Gallant-White cosine-smasher $f(x) = \begin{cases} 0 & x \leq -\frac{\pi}{2} \\ \frac{1+cos(\frac{3}{2}\pi)}{2} & x \in [-\frac{\pi}{2}, \frac{\pi}{2}] \\ 0 & x \geq \frac{\pi}{2} \end{cases}$.

Remark 1. Note that NNREACH$(id, e^{(\cdot)})$ is equivalent to CSP$(\mathbb{R}; R_+, R., 1, e^{(\cdot)})$, also known as Tarski's exponential function problem. It is not even known whether this problem is decidable or not (cf. [17]). Similar approaches have recently been made in [11].

4 Network Equivalence and Verification of Interval Property

In this section, we study the complexity of the remaining decision problems VIP and NE introduced in Definition 2. We will see that in a lot of cases these problems are essentially the same.

Theorem 5. *Let F be a set of activation functions such that sign, $id \in F$.*
a) NE(F) *one-one reduces to the complement of* NNREACH(F) *in linear time. Consequently,* NE*(id) is in P.*
b) NNREACH(F) *one-one reduces to the complement of* NE(F) *in linear time. Consequently,* NE*(ReLU) is co-NP-complete and* NE*(f) is co-NP-hard for any non-linear f that is continuous on an interval.*
c) NE *truth-table reduces to* NE *with just one output dimension in linear time independent of the set of activation functions.*
The same holds for Heaviside or a similar step function instead of sign, id can independently be replaced by ReLU.

Theorem 6. *Let F be a set of activation functions.*
a) VIP(F) *truth-table reduces to the complement of* NNREACH(F) *in linear time. Consequently,* VIP*(id) is in P and* VIP*(ReLU) is in co-NP.*
b) Let F contain at least one among the functions H (the Heaviside function), sign or ReLU. Then NNREACH(F) *one-one reduces to the complement of* VIP(F) *in linear time.*
c) VIP(F) *truth-table reduces to* VIP(F) *with just one output condition in linear time, meaning the output constraint is just one strict inequality.*

Theorem 7. *Let F be a set of activation functions containing id or ReLU and H or sign.*
 a) $\mathrm{NE}(F)$ *one-one reduces to* $\mathrm{VIP}(F)$ *in linear time.*
 b) $\mathrm{VIP}(F)$ *one-one reduces to* $\mathrm{NE}(F)$ *in linear time.*

5   Conclusion and Further Questions

We examined the computational complexity of the reachability problem for neural networks in dependence of the activation functions used. We provided conditions for includedness and hardness for NP, translated a dichotomy result for certain CSPs into the language of reachability problems, and showed ETR-hardness of the reachability problems for several typical activation functions. We also showed that NE and VIP are in many cases the same problem which is essentially "co-NNREACH". Further open questions are:

1.) Do there exist activation functions that are not semilinear but still lead to a reachability problem in NP?
2.) Can the reachability problem for the sigmoid function $f(x) = 1/(1 + e^{-x})$, one oft the most frequently used activations, be classified any better than just as NP-hard? Can it be related to ETR?
3.) Can the discussed ETR-hard problems be classified with respect to (potentially) larger complexity classes such as PSPACE, EXP-Time, decidable,...?
4.) Can reductions between VIP, NE and NNREACH be found that do not rely on certain functions to be included in the set of activations?
5.) How do these problems behave when the deciding algorithm is considered as a computation model with reals as entities? For example of which complexity are the respective problems in a model of real computations like the Blum-Shub-Smale model [1]? What if the underlying neural net may have any real weights and biases instead of just rational ones?

References

1. Blum, L., Cucker, F., Shub, M., Smale, S.: Complexity and Real Computation. Springer, New York (1998). https://doi.org/10.1007/978-1-4612-0701-6
2. Bodirsky, M., Jonsson, P., von Oertzen, T.: Essential convexity and complexity of semi-algebraic constraints. Log. Methods Comput. Sci. **8**, 1–25 (2012)
3. Bulatov, A.A., Jeavons, P., Krokhin, A.: Classifying the complexity FO constraints using finite algebras. SIAM J. Comput. **34**, 720–742 (2003)
4. Calin, O.: Deep Learning Architectures - A Mathematical Approach. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-36721-3
5. Canny, J.: Some algebraic and geometric computations in PSPACE. Computer Science Division, University of California, Berkeley, Technical report (1988)

6. Carbonnel, C., Cooper, M.C.: Tractability in constraint satisfaction problems: a survey. Assoc. Comput. Mach. **21**, 115–144 (2016)
7. Dixon, M., Klabjan, D., Bang, J.H.: Classification-based financial markets prediction using deep neural networks. Alg. Financ. **6**, 67–77 (2017)
8. Grigorescu, S., Trasnea, B., Cocias, T., Macesanu, G.: A survey of deep learning techniques for autonomous driving. J. Field Robot. **37**, 362–386 (2019)
9. Hinton, G., et al.: Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Process **29**, 82–97 (2012)
10. Huang, X., et al.: A survey of safety and trustworthiness of deep neural networks: verification, testing, adversarial attack and defence, and interpretability. Comput. Sci. Rev. **37**, 100270 (2020)
11. Isac, O., Zohar, Y., Barrett, C., Katz, G.: DNN Verification, Reachability, and the Exponential Function Problem (2023)
12. Karmarkar, N.: A new polynomial-time algorithm for linear programming. Combinatorica **4**, 373–396 (1984)
13. Katz, G., Barrett, C., Dill, D., Julian, K., Kochenderfer, M.: Reluplex: an efficient SMT solver for verifying deep neural networks. Comput. Aided Verif. **10426**, 97–117 (2017)
14. Khan, A., Sohail, A., Zahoora, U., Qureshi, A.S.: A survey of the recent architectures of deep convolutional neural networks. Artif. Intell. Rev. **53**(8), 5455–5516 (2020). https://doi.org/10.1007/s10462-020-09825-6
15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Assoc. Comput. Mach. **25**, 1–9 (2017)
16. Litjens, G., Kooi, T., Bejnordi, B.E., Setio, A.A.A., Ciompi, F., Ghafoorian, M., van der Laak, J.A., van Ginneken, B., Sánchez, C.I.: A survey on deep learning in medical image analysis. Med. Image Anal. **42**, 60–88 (2017)
17. Macintyre, A., Wilkie, A.: On the Decidability of the Real Exponential Field. Kreiseliana: About and Around Georg Kreisel, pp. 451–477 (1996)
18. Ruan, W., Huanga, X., Kwiatkowska, M.: Reachability analysis of deep neural networks with provable guarantees. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI, pp. 2651–2659 (2018)
19. Schaefer, M., Štefankovič, D.: Fixed points, Nash equilibria, and the existential theory of the reals. Theory Comput. Syst. **60**, 172–193 (2012)
20. Sälzer, M., Lange, M.: Reachability is NP-complete even for the simplest neural networks. Int. Conf. Reachabi. Probl. **13035**, 149–164 (2021)