

GOETHE UNIVERSITÄT

FACHBEREICH 12

Tropical Geometry of Deep Neural Networks

Author:

David LEATHAM (6087632)

Lecturer:

Prof. Dr. Martin ULIRSCH

Submission Date: September 21, 2020



Contents

0	Introduction	2
1	Tropical Algebra	4
2	Neural networks	12
3	Tropical algebra of neural networks	22
4	Tropical hypersurfaces	27
5	Tropical geometry of neural networks	33
6	Conclusion	40

0 Introduction

Tropical geometry is a modern branch of mathematics subject to a rise in research throughout the past 30 years. Neural networks have found application in many disciplines and have been used to algorithmically manage complex tasks like self driving cars, face and text recognition or automatically processing hand written checks at banks. Neural networks impact our lives on a daily basis.

In this thesis we describe a new representation of neural networks through tropical geometry, following the recent article "Tropical Geometry of Deep Neural Networks" by Liwen Zhang, Gregory Naitzat and Lek-Heng Lim [ZNL18].

The subject of neural networks was introduced by Warren McCulloch and Walter Pitts in 1943 when they created a computational model for neural networks [MP43]. In the following years, one of the next big steps towards today's state of the art neural networks was the research of the underlying concepts of backpropagation in 1960 by Kelley [Kel60] and in 1961 by Bryson [Bry61]. Backpropagation is an algorithm describing a key concept of training neural networks. Starting around 2010 to this day, neural networks are experiencing a boost in popularity. This is due to hardware improvements especially in GPU's making backpropagation feasible [CMGS10], the growing amount of data and the improvement of knowledge on deep neural networks fuelled by articles and works, for example that of Andrew Yan-Tak Ng and Jeff Dean in 2012. They created a network that learned to recognize higher-level concepts, such as cats, only from watching unlabeled images [LMD⁺11]. Nowadays neural networks help solve challenging problems in computer science and software engineering and, in particular, they are key to application areas such as system identification and control, pattern and sequence recognition, social network filtering and e-mail spam filtering.

Tropical geometry on the other hand is a field of interest in mathematics, whose underlying analytical ideas have been around some years before 1990, for example by Victor Pavlovich Maslov [Mas85]. But only since then have there been basic efforts to consolidate the basic definitions as research increased and thus, from 2000 onwards, gained the attention it receives nowadays, carried by successful applications in enumerative tropical algebraic geometry for instance by Grigory Mikhalkin in 2005 [Mik05]. Tropical geometry is the study of polynomials under tropical multiplication and addition, where tropical multiplication is usual addition and tropical addition of two elements returns the minimal element. It is used in mathematics itself [Kri14], finance [Kle09] and computer science. For the last topic this thesis is an example as we describe a fundamental mathematical connection between tropical geometry and deep neural networks.

Concerning the structure of the thesis, we start by introducing the reader to the necessary theories of tropical geometry in section 1 and the theory of neural networks in section 2, but also extend the exploration of the topics beyond the scope needed for the fundamental Theorems in this thesis, giving a more rounded introduction to these topics themselves when appropriate. Section 3 applies the knowledge of the first two sections to demonstrate a fundamental link between neural networks and the difference between two tropical polynomials, called tropical rational functions, which is the core insight of

the thesis. Then, to put this knowledge to use, we formulate statements about decision boundaries of neural networks in section 5 with tropical geometry, or more concrete, the link between neural networks and tropical rational functions. This needs fundamental knowledge of tropical hypersurfaces that are introduced in section 4.

This thesis follows the underlying paper "Tropical Geometry of Deep Neural Networks" by Liwen Zhang, Gregory Naitzat and Lek-Heng Lim very closely, in that all main statements may be reordered and rewritten, but come directly out of [ZNL18] and are cited accordingly.

1 Tropical Algebra

The remaining sections will build upon section 1 without exception, therefore we begin by introducing tropical algebra. The main goal is to establish an intuition and formal understanding of subsets of functions we call tropical and tropical rational functions, if the image is one dimensional, or tropical and tropical rational maps if not.

Our basic object of study is the semifield $(\mathbb{R} \cup \{-\infty\}, \oplus, \odot)$. As a set this corresponds to the real numbers \mathbb{R} , together with an extra element $-\infty$ which represents minus infinity. In this semifield both operations are defined on two elements $x, y \in \mathbb{R} \cup \{-\infty\}$. The first operation is called the **tropical sum** and returns the maximum of two elements, while the second operation is called **tropical product** and returns the usual sum:

$$x \oplus y := \max(x, y) \quad \text{and} \quad x \odot y := x + y.$$

To be in a position to understand the properties of $(\mathbb{R} \cup \{-\infty\}, \oplus, \odot)$, we define a semiring and a semifield formally.

Definition 1.1. [BP85] A **semiring** is a set \mathcal{R} equipped with two binary operations $+$ and \cdot , called addition and multiplication, such that for $a, b, c \in \mathcal{R}$

- $(\mathcal{R}, +)$ is a commutative monoid with identity element 0, meaning the following holds:
 - $(a + b) + c = a + (b + c)$
 - $0 + a = a + 0 = a$
 - $a + b = b + a$
- (\mathcal{R}, \cdot) is a monoid with identity element 1, meaning:
 - $(a \cdot b) \cdot c = a \cdot (b \cdot c)$
 - $1 \cdot a = a \cdot 1 = a$
- Multiplication left and right distributes over addition:
 - $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$
 - $(a + b) \cdot c = (a \cdot c) + (b \cdot c)$
- Multiplication by 0 annihilates:
 - $0 \cdot a = a \cdot 0 = 0$

Proposition 1.2. [MS15, p. 10] The object $(\mathbb{R} \cup \{-\infty\}, \oplus, \odot)$ is a semiring.

Proof. The neutral element for the tropical sum is $-\infty$ since for $x \in \mathbb{R} \cup \{-\infty\}$, $x \oplus -\infty = \max(x, -\infty) = x$ holds and with $x \odot 0 = x + 0 = x$ for $x \in \mathbb{R}$, 0 is the neutral element of tropical multiplication. Both tropical addition and multiplication are commutative. This is obvious for $x, y \in \mathbb{R}$, so take $y \in \mathbb{R} \cup \{-\infty\}$ and with the following all cases are shown:

$$\begin{aligned} -\infty \oplus y &= \max(-\infty, y) = y = \max(y, -\infty) = y \oplus -\infty \\ -\infty \odot y &= -\infty + y = \infty = y + -\infty = y \odot -\infty. \end{aligned}$$

Tropical multiplication distributes over addition since if we take $x, y, z \in \mathbb{R}$ then

$$\begin{aligned} x \odot (y \oplus z) &= x + \max(y, z) = \max(x + y, x + z) = (x \odot y) \oplus (x \odot z) \\ (y \oplus z) \odot x &= \max(y, z) + x = \max(y + x, z + x) = (y \odot x) \oplus (z \odot x) \end{aligned}$$

holds. Multiplication by $-\infty$ annihilates $-\infty \odot x = -\infty \forall x \in \mathbb{R} \cup \{-\infty\}$ and therefore $(\mathbb{R} \cup \{-\infty\}, \oplus, \odot)$ is a semiring. \square

Definition 1.3. [OGAJ20] A commutative semiring in which every nonzero element is multiplicatively invertible is called a **semifield**.

Tropical multiplication corresponds to usual addition which is commutative and invertible for all $x \in \mathbb{R}$. The inverse operation to addition is subtraction denoted as, \ominus , **tropical division**. As shown in 1.2 the elements $-\infty$ represents zero. This shows the following proposition.

Proposition 1.4. The object $(\mathbb{R} \cup \{-\infty\}, \oplus, \odot)$ is a semifield.

Definition 1.5. We call the semifield $\mathbb{T} := (\mathbb{R} \cup \{-\infty\}, \oplus, \odot)$ the **tropical semifield**.

In tropical geometry often infinity instead of minus infinity and min instead of max are used. This does not change any of the underlying theories of tropical algebra as the two semirings are tropically isomorph. This means

$$\begin{aligned} (\mathbb{R} \cup \{-\infty\}, \oplus := \max, \odot) &\rightarrow (\mathbb{R} \cup \{\infty\}, \oplus := \min, \odot), \\ x &\mapsto \begin{cases} -x & \text{for } x \in \mathbb{R} \\ \infty & \text{for } x = -\infty \end{cases} \end{aligned}$$

is an Isomorphism of semifields.

An essential feature of tropical arithmetics is that there is no subtraction. Take $a, b \in \mathbb{R} \cup \{-\infty\}$ with $a > b$ then the equation $a \oplus x = b$ has no solution x at all. [MS15, p. 11]

To get more familiar with the tropical arithmetic, we will have a look at an example before we introduce tropical polynomials.

Remark 1.6. We will examine the tropical Pascal's triangle, whose rows are the coefficients appearing in a binomial expansion [MS15], which means that the n -th layer of the triangle consists of $n + 1$ entries corresponding to p_0, \dots, p_n in:

$$(a \oplus b)^n = (p_0 \odot a^n) \oplus (p_1 \odot a^{n-1} \odot b) \oplus \dots \oplus (p_{n-1} \odot a \odot b^{n-1}) \oplus (p_n \odot b^n).$$

for $a, b \in \mathbb{T}$. Let us calculate the entries for the fourth row and set $a, b \in \mathbb{T}$:

$$\begin{aligned} (a \oplus b)^3 &= 3 \max(a, b) \\ &= \max(3a, 3b) = (a^3 + b^3) \\ &= \max(0 \odot a^3, 0 \odot a^2 \odot b, 0 \odot a \odot b^2, 0 \odot b^3) \\ &= (0 \odot a^3) \oplus (0 \odot a^2) \odot (b \oplus 0 \odot a) \odot (b^2 \oplus 0 \odot b^3). \end{aligned}$$

You may say the Pascal's coefficients are four zeros. And actually the same applies to all cases:

$$\begin{aligned}(a \oplus b)^n &= a^n \oplus b^n \\ &= (0 \odot a^n) \oplus (0 \odot a^{n-1} \odot b) \oplus \cdots \oplus (0 \odot a \odot b^{n-1}) \oplus (0 \odot b^n).\end{aligned}$$

Therefore the tropical Pascal's triangle, whose rows are the coefficients appearing in a binomial expansion, take a simple shape. All its coefficients are zero.

$$\begin{array}{cccccccc}n = 0 & & & & & & & 0 \\n = 1 & & & & 0 & & 0 & \\n = 2 & & & 0 & & 0 & & 0 \\n = 3 & & & 0 & & 0 & & 0 \\n = 4 & & 0 & & 0 & & 0 & & 0 \\n = 5 & & 0 & & 0 & & 0 & & 0 \\n = 6 & 0 & & 0 & & 0 & & 0 & & 0.\end{array}$$

Moving on, first, so that we can introduce multidimensional tropical polynomials properly, a notion of monomials is needed.

Definition 1.7. [ZNL18, p. 2] A **tropical monomial** in d variables x_1, \dots, x_d is a function $f : \mathbb{T}^d \rightarrow \mathbb{T}$ of the form

$$f(x_1, \dots, x_d) = c \odot x_1^{a_1} \odot x_2^{a_2} \odot \cdots \odot x_{d-1}^{a_{d-1}} \odot x_d^{a_d}$$

where $c \in \mathbb{R} \cup \{-\infty\}$ and $a_1, \dots, a_d \in \mathbb{N}$. As a convenient shorthand notation, we will also write a tropical monomial in multiindex notation as cx^α where $\alpha = (a_1, \dots, a_d) \in \mathbb{N}_d$ and $x = (x_1, \dots, x_d)$. Note that $x^\alpha = 0 \odot x^\alpha$.

Definition 1.8. [ZNL18, p. 2] A **tropical polynomial** $f : \mathbb{T}^d \rightarrow \mathbb{T}$ is a finite tropical sum of tropical monomials

$$f(x) = c_1 x^{\alpha_1} \oplus \cdots \oplus c_r x^{\alpha_r}$$

where $\alpha_i = (\alpha_{i1}, \dots, \alpha_{id}) \in \mathbb{N}^d$ and $c_i \in \mathbb{T}$, $i = 1, \dots, r$. We will assume that a monomial of a given multiindex appears at most once in the sum, i.e. $\alpha_i \neq \alpha_j$ for any $i \neq j$.

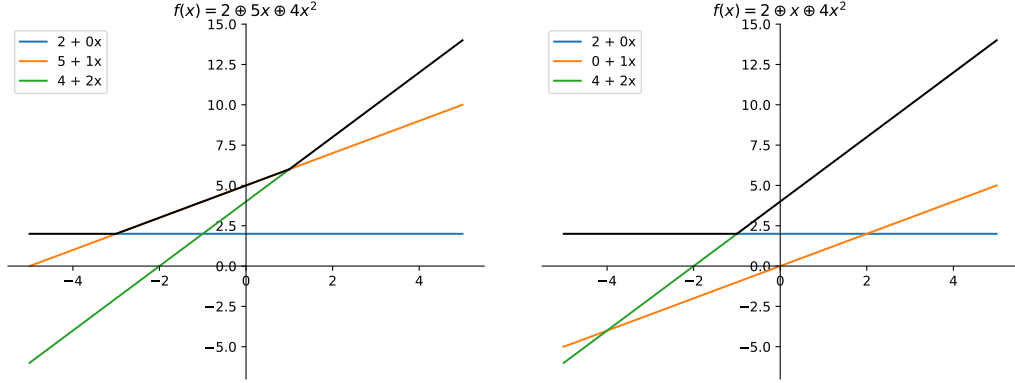
The condition $\alpha_i \neq \alpha_j$ with constant c_k 's for $k = 1, \dots, r$ does not restrict the tropical polynomial, but only standardises the form of representation since for $\alpha_i = \alpha_j$ the two monomials in f can be combined to one monomial without changing the function as follows

$$\begin{aligned}c_j x^{\alpha_j} \oplus c_i x^{\alpha_j} &= \max\{c_j + \alpha_j x, c_i + \alpha_i c_i\} \\ &= \begin{cases} c_j x^{\alpha_j} & \text{for } c_j \geq c_i \\ c_i x^{\alpha_i} & \text{for } c_j < c_i. \end{cases}\end{aligned}$$

Remark 1.9. Now we examine tropical polynomials in one variable. Monomials in one variable are of the form $c \odot x^a : \mathbb{T} \rightarrow \mathbb{T}$ where $c \in \mathbb{T}$ and $a \in \mathbb{N}$, which means a tropical polynomial in one variable is of the form $f(x) = c_1 x^{\alpha_1} \oplus \cdots \oplus c_r x^{\alpha_r}$ with $c_i \in \mathbb{T}$, $a_i \in$

\mathbb{N} for $i = 1, \dots, r$.

For a quadratic tropical polynomial $f(x) = a \oplus bx \oplus cx^2$ with $a, b, c \in \mathbb{T}$ linearity breaks at two points $b - c$ and $a - b$ if the condition $b - c > a$ holds and otherwise only breaks at one point $\frac{a-c}{2}$. Visualising two quadratic tropical functions gives a good intuition of the piecewise-linearity. The black coloured parts of figures 1a and 1b indicate the graph of the tropical polynomial.



- (a) Take $f(x) = 2 \oplus 5x \oplus 4x^2$ a quadratic polynomial. The graph equals $\max(2, 5 + x, 4 + 2x)$. Until $5 - 2$, 2 dominates, from there $5 + x$ dominates to the point $4 + 2x$.
- (b) Take $f(x) = 2 \oplus x \oplus 4x^2$, we have changed the scalar in the second part. This way it plays no role in the tropical polynomial.

Figure 1: Quadratic tropical polynomials.

We can see the degree of a tropical polynomial, defined the same as a degree of a usual polynomial, gives an upper bound for the number of non-linear edges of the tropical polynomial, but not the exact value. With higher degree polynomials more non-linear edges are possible as figure 2 illustrates.

Definition 1.10. Let S be a set in a real vector space. Then S is **convex** if for $s_1, s_2 \in S$:

$$\lambda s_1 + (1 - \lambda)s_2 \in S$$

holds for all $\lambda \in [0, 1] \subset \mathbb{R}$.

Definition 1.11. Let X be a convex set in a real vector space and let $f : X \rightarrow \mathbb{R}$ be a function, then f is called **convex** if for $x_1, x_2 \in X$ and $\lambda \in [0, 1] \subset \mathbb{R}$, $f(\lambda x_1 + (1 - \lambda)x_2) \leq \lambda f(x_1) + (1 - \lambda)f(x_2)$ holds.

Definition 1.12. A function $f : \mathbb{R}^m \rightarrow \mathbb{R}^n$ with $n, m \in \mathbb{N}$ is **piecewise linear** if there exist closed sets $(P_i)_{i \in \mathbb{N}} \subset \mathbb{R}^m$ with $f : P_i \rightarrow \mathbb{R}^n$ linear and $\cup_i P_i = \mathbb{R}^m$.

In multiple variables we can characterise tropical polynomials as functions from $\mathbb{R}^n \rightarrow \mathbb{R}$ that satisfy the following three properties.

Lemma 1.13. [MS15] Let f be a tropical polynomial

$$f(x) = c_1 x^{\alpha_1} \oplus \dots \oplus c_r x^{\alpha_r}$$

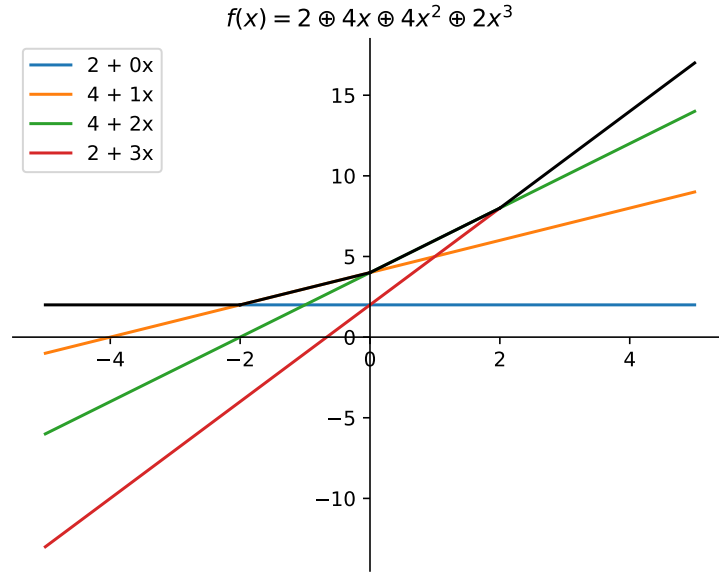


Figure 2: Take $f(x) = 2 \oplus x \oplus 4x^2 \oplus 3x^3$. For a degree three tropical polynomial in one variable this polynomial has reached the maximum number of non linear edges.

as in definition 1.8, but we restrict $c_1, \dots, c_r \in \mathbb{R}$ to be real and $f : \mathbb{R}^n \rightarrow \mathbb{R}$ with $n \in \mathbb{N}$. Then f has three important properties:

- (1) f is continuous.
- (2) f is piecewise-linear, where the number of pieces is finite.
- (3) f is convex, i.e. $p(\frac{x+y}{2}) \leq \frac{1}{2}(p(x) + p(y)) \forall x, y \in \mathbb{R}$

Proof. The minimum of continuous functions is still continuous which proves (1). Every monomial $c_i x^{\alpha_i} = c_i + x_1 \alpha_{i1} + \dots + x_n \alpha_{in}$ is per definition linear. Because of (1), linearity can only be broken where $c_i x^{\alpha_i} = c_j x^{\alpha_j}$ for $i \leq j$ and $i, j = 1, \dots, r$. A piece is a single maximal region of $c_i x^{\alpha_i}$ on which linearity is not broken. If we introduce $c_i x^{\alpha_i}$ one after another, then in the i -th step not more than i^2 new pieces can be created, so there can

only be $\sum_{i=1}^r i^2$ or less pieces which proves (2). For (3) let $x, y \in \mathbb{R}^n$, then:

$$\begin{aligned}
& f(\lambda x + (1 - \lambda)y) \\
&= \bigoplus_{i=1}^r c_i (\lambda x + (1 - \lambda)y)^{\alpha_i} \\
&= \max_{i=1}^r \{c_i + \sum_{j=1}^n \alpha_{ij} (\lambda x_j + (1 - \lambda)y_j)\} \\
&= \max_{i=1}^r \{\lambda(c_i + \sum_{j=1}^n \alpha_{ij} x_j) + (1 - \lambda)(c_i + \sum_{j=1}^n \alpha_{ij} y_j)\} \\
&\leq \max_{i=1}^r \{\lambda(c_i + \sum_{j=1}^n \alpha_{ij} x_j)\} + \max_{i=1}^r \{(1 - \lambda)(c_i + \sum_{j=1}^n \alpha_{ij} y_j)\} \\
&= \lambda \max_{i=1}^r \{c_i + \sum_{j=1}^n \alpha_{ij} x_j\} + (1 - \lambda) \max_{i=1}^r \{c_i + \sum_{j=1}^n \alpha_{ij} y_j\} \\
&= \lambda f(x) + (1 - \lambda)f(y)
\end{aligned}$$

which ends the prove. \square

The same could be argued for the tropical function f when $c_i \in \mathbb{T}$ and $f : \mathbb{T}^n \rightarrow \mathbb{T}$, but the underlying theories, for example for linearity, change for semifields, which for example becomes clear in [OGAJ20]. In the rest of the thesis we will refer to tropical polynomials as piecewise-linear in general and only use linearity in the real case. By restricting the domain to real numbers the image must consequently also be real.

Proposition 1.14. Every function $\mathbb{R}^n \rightarrow \mathbb{R}$, which satisfies the three properties (1), (2) and (3) from lemma 1.13, has a representation as the minimum of a finite set of linear functions. Thus, the tropical polynomials in n variables x_1, \dots, x_n represent the class of piecewise-linear convex functions on \mathbb{R}^n with integer coefficients.

Proof. This follows directly from lemma 1.13. \square

We are now able to introduce tropical rational functions and the semifields $\mathbb{T}[X_1, \dots, X_d]$, $\mathbb{T}(X_1, \dots, X_d)$. These are important to understand the core section (section 3), in particular the actual connection build between neural networks and tropical geometry in this thesis.

Definition 1.15. [ZNL18, p. 3] A **tropical rational function** is a standard difference, or, equivalently, a tropical quotient of two tropical polynomials $f(x)$ and $g(x)$:

$$(f - g)(x) = f(x) - g(x) = f(x) \oslash g(x) = (f \oslash g)(x)$$

Proposition 1.16. [ZNL18, p. 3] The sets

$$\begin{aligned}
\mathbb{T}[X_1, \dots, X_d] &:= \{f : \mathbb{T}^d \rightarrow \mathbb{T} \mid f \text{ is tropical polynomial}\} \text{ and} \\
\mathbb{T}(X_1, \dots, X_d) &:= \{f : \mathbb{T}^d \rightarrow \mathbb{T} \mid f \text{ is tropical rational function}\}
\end{aligned}$$

are both semirings.

Proof. First we show, that for two tropical polynomials $f, g \in \mathbb{T}[X_1, \dots, X_r]$ with $f(x) = \bigoplus_{i=0}^{\alpha} a_i x^{\zeta_i}$ and $g(x) = \bigoplus_{i=0}^{\beta} b_i x^{\omega_i}$ their tropical multiplication is a tropical polynomial $f \odot g \in \mathbb{T}[X_1, \dots, X_r]$,

$$\begin{aligned}
(f \odot g)(x) &= f(x) + g(x) \\
&= \left(\bigoplus_{i=0}^{\alpha} a_i x^{\zeta_i} \right) + \left(\bigoplus_{i=0}^{\beta} b_i x^{\omega_i} \right) \\
&= \max_{i=0}^{\alpha} \left\{ a_i + \sum_{k=1}^r \zeta_{ik} x_k \right\} + \max_{j=0}^{\beta} \left\{ b_j + \sum_{k=1}^r \omega_{jk} x_k \right\} \\
&= \max_{\substack{i=1, \dots, \alpha \\ j=1, \dots, \beta}} \left\{ a_i + \sum_{k=1}^r \zeta_{ik} x_k + b_j + \sum_{k=1}^r \omega_{jk} x_k \right\} \\
&= \max_{\substack{i=1, \dots, \alpha \\ j=1, \dots, \beta}} \left\{ (a_i + b_j) + \sum_{k=1}^r (\zeta_{ik} + \omega_{jk}) x_k \right\} \\
&= \bigoplus_{\substack{i=1, \dots, \alpha \\ j=1, \dots, \beta}} (a_i \odot b_j) \odot x^{\zeta_i + \omega_j} \in \mathbb{T}[X_1, \dots, X_r].
\end{aligned}$$

That $(f \oplus g) \in \mathbb{T}[X_1, \dots, X_r]$ is immediate. We have already shown that \mathbb{T} is a tropical semifield. Therefore, except for the neutral elements, all other axioms hold automatically pointwise.

The neutral element for the tropical sum is $-\infty = -\infty \odot x = x \odot -\infty$ $x \in \mathbb{T}$. Both $0, -\infty \in \mathbb{T}$ are also a tropical function and a tropical rational function. Since for $f(x) \in \mathbb{T}(X_1, \dots, X_r)$ or $f(x) \in \mathbb{T}[X_1, \dots, X_r]$, the following holds

$$\begin{aligned}
f(x) \oplus \infty &= \max(f(x), -\infty) = f(x) \\
f(x) \odot 0 &= f(x) + 0 = f(x).
\end{aligned}$$

Therefore 0 is the neutral element of tropical multiplication for tropical and tropical rational functions and $-\infty$ is the neutral element of addition. We have shown $\mathbb{T}[X_1, \dots, X_r]$ is a semifield.

Let $g, f, h \in \mathbb{T}(X_1, \dots, X_d)$ with

$$\begin{aligned}
f(x) &= f_1(x) \odot f_2(x) \\
g(x) &= g_1(x) \odot g_2(x) \\
h(x) &= h_1(x) \odot h_2(x)
\end{aligned}$$

The tropical sum of tropical rational functions is a tropical rational function:

$$\begin{aligned}
(f \oplus g)(x) &= f(x) \oplus g(x) \\
&= (f_1(x) \odot f_2(x)) \oplus (g_1(x) \odot g_2(x)) \\
&= \min\{f_1(x) - f_2(x), g_1(x) - g_2(x)\} \\
&= \min\{f_1(x) + g_2(x), g_1(x) + f_2(x)\} - f_2(x) - g_2(x) \\
&= (f_1(x) + g_2(x) \oplus g_1(x) + f_2(x)) \\
&\quad \odot (f_2(x) + g_2(x)) \in \mathbb{T}(X_1, \dots, X_r),
\end{aligned}$$

since addition as tropical addition of tropical polynomials is a tropical polynomial. The tropical product of tropical rational functions is a tropical rational function:

$$\begin{aligned}
(f \odot g)(x) &= f(x) \odot g(x) \\
&= (f_1(x) \odot f_2(x)) \odot (g_1(x) \odot g_2(x)) \\
&= f_1(x) - f_2(x) + g_1(x) - g_2(x) \\
&= (f_1(x) + g_1(x)) - (f_2(x) + g_2(x)) \in \mathbb{T}(X_1, \dots, X_r)
\end{aligned}$$

□

Proposition 1.17. The set $\mathbb{T}(X_1, \dots, X_d)$ is a semifield.

Proof. Because of proposition 1.16 all we have to show is that $f \oslash g \in \mathbb{T}(X_1, \dots, X_d)$ has a multiplicative inverse element. With $(f \oslash g) \odot (g \oslash f) = f - g + g - f = 0$ we found $(g \oslash f) \in \mathbb{T}(X_1, \dots, X_d)$ the element. □

We regard a tropical polynomial $f = f \oslash 0$ as a special case of a tropical rational function and thus $\mathbb{T}[X_1, \dots, X_r] \subseteq \mathbb{T}(X_1, \dots, X_r)$. [ZNL18, p. 3]

Remark 1.18. A d-variate tropical polynomial $f(x)$ defines a function $f : \mathbb{T}^d \rightarrow \mathbb{T}$ that is a convex function in the usual sense as taking max and \sum of convex functions preserve convexity [BBV04].

Definition 1.19. [ZNL18, p. 3] $F : \mathbb{R}^d \rightarrow \mathbb{R}^p, x = (x_1, \dots, x_d) \mapsto (f_1(x), \dots, f_p(x))$, is called a **tropical polynomial map** if each $f_i : \mathbb{R}^d \rightarrow \mathbb{R}$ is a tropical polynomial, $i = 1, \dots, p$, and a **tropical rational map** if f_1, \dots, f_p are tropical rational functions. We will denote the set of tropical polynomial maps by $Pol(d, p)$ and the set of tropical rational maps by $Rat(d, p)$. So $Pol(d, 1) = \mathbb{T}[X_1, \dots, X_d]$ and $Rat(d, 1) = \mathbb{T}(X_1, \dots, X_d)$.

2 Neural networks

Historically the term "Neural Network" was introduced in attempts to describe the functionality of biological processes, in particular the nervous system and the brain, in a mathematical sense [MP43, WH60, RHW86]. Simply put the nervous system is a net of neurons, each having a soma and an axon. "At any instant a neuron has some threshold, which excitation must exceed to initiate an impulse. This, except for the fact and the time of its occurrence, is determined by the neuron, not by the excitation. From the point of excitation the impulse is propagated to all parts of the neuron." [MP43, p. 1] Through synapses the axons are connected to the soma of further neurons, this way the impulse is passed on. Impulses passing through the nervous system partly consist of electrical impulses and chemical reactions [Pal56]. A collection of partially connected neurons, capable of carrying impulses, is called a biological neural network.

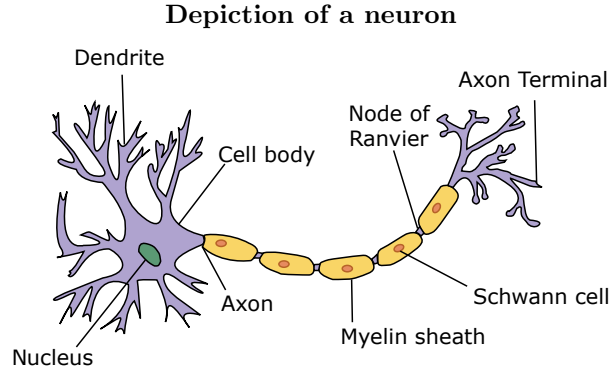


Figure 3: A representation of a neuron. The axon terminal attach to the dendrite of another neuron. This way impulses pass from neuron to neuron [Jar].

Based on a biological neural network, an abstract neuronal network, in the following simply referenced to as neural network, is defined. Biological neurons and especially biological neural networks are far more complex than this introduction may make it seem. Because of its vast complexity, biological realism would impose entirely unnecessary constraint when building a fast computable model. Boiling a biological neuronal network down to its quintessential features results in a weighted directed graph, where edges and vertices are weighted. Typically a weighted graph only has its edges weighted. To fit our model we introduce them also with weighted vertices.

Definition 2.1. [BW10, p. 148] A **finite graph** or simply **graph** is a pair $G = (V, E)$, where V is a finite set whose elements are called vertices, and $E \subset \{\{x, y\} | (x, y) \in V \times V\}$ is a set of two-sets of vertices, whose elements are called edges.

A small tweak to this definition yields the definition of a directed graph.

Definition 2.2. A **directed graph** is a pair $G = (V, E)$, where V is a set whose elements are called vertices, and $E \subset \{(x, y) | (x, y) \in V \times V\}$ is a set of edges which are ordered pairs of distinct vertices.

Definition 2.3. Let \mathbb{K} be a field. A **vertex weighted graph** $G = (V, E, \psi)$ is a graph with a function $\psi : V \rightarrow \mathbb{K}$ that assigns a weight $\psi(v)$ to each vertex $v \in V$. An **edge weighted graph** $G = (V, E, \omega)$ is a graph G with a function $\omega : E \rightarrow \mathbb{K}$ that assigns a weight $\omega(e)$ to each edge $e \in E$. A **weighted graph** $G = (V, E, \psi, \omega)$ or short $G = (V, E)$ is a graph with weighted edges and vertices.

Graphs can represent a multitude of relations between objects, like road maps of roads connecting cities, or describe objects themselves like molecules.

Example 2.4. For instance, to introduce graphs and demonstrate a connection between tropical geometry and graph theory, the edge weighted directed Graph $G = (V, E, \omega)$, depicted in figure 4, could depict a map of hiking trails which connect huts as nodes. The trails are weighted with the distance between huts and the goal is to find the minimal walking distance from hut A to any other. We will use the opportunity to introduce a specific graph formally. Matching figure 4 the graph is fully defined by setting $V = \{A, B, C, D, E, F, G, H\}$, $E = \{(A, B), (A, H), (A, E), (B, D), (B, C), (B, H), (C, D), (C, G), (D, G), (E, H), (E, F), (E, G), (F, C), (F, G), (H, C), (H, F)\}$ and in the same order as we have written the edges of E here, the values of ω correspond to 5, 8, 9, 15, 12, 4, 3, 11, 9, 5, 4, 20, 1, 13, 7, 6.

Min distance weighted graph

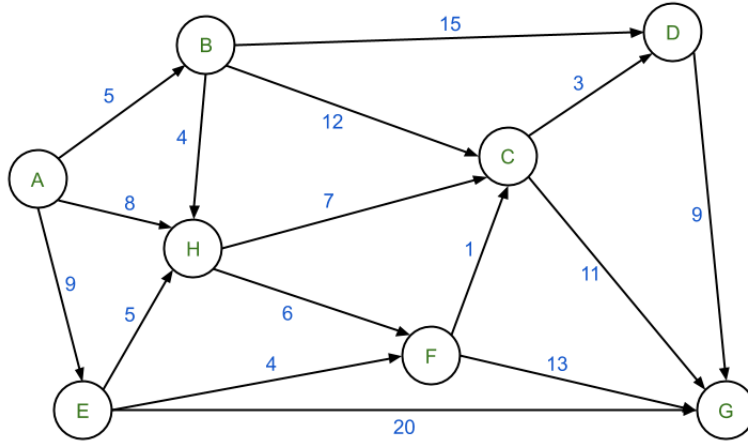


Figure 4: Min distance weighted graph [Ale].

To compute an answer we use Dijkstra's algorithm as, for example, introduced in [BG14]. At any point in the algorithm simply consider all reachable nodes from nodes that have their shortest path already computed by adding up the computed shortest paths length to the paths length of the reachable node. This way every reachable node by nodes with their longest path already computed get at least one or multiple lengths of paths assigned to them. Pick the shortest path under all of these and repeat until all vertices have a path assigned. If there are multiple shortest paths, pick one of them. At

any point, if there are no more reachable nodes, terminate and the nodes that have no distance assigned are not reachable.

Proof. Let $v_0 \in V$ be the starting vertex and v_1 an adjacent vertex with the closest path to v_0 . Say $\text{dist}(v)$ is the computed shortest distance from node v_0 to $v \in V$, then $\text{dist}(v_1) = \omega((v_0, v_1))$. Let $n \in \mathbb{N}$, now say v_1, \dots, v_n are the n nodes with the shortest paths $\text{dist}(v_1), \dots, \text{dist}(v_n)$ and $v_{n+1} \in V \setminus v_1, \dots, v_n$ the vertex of closest distance to one of $v_1 \dots v_n$ out of all adjacent vertices. We pick $i \in \{1, \dots, n\}$ so that v_i is the vertex to which v_{n+1} is closest, then $\text{dist}((v_1, v_{n+1})) = \text{dist}((v_1, v_n)) + \omega((v_n, v_{n+1}))$ since otherwise v_{n+1} would have been under the first n closest vertices. Also v_1, \dots, v_{n+1} are the $n + 1$ closest vertices to v_0 \square

Using dijkstra's algorithm it is easy to compute the shortest distance from node A to D is 17 for example.

An interesting link to tropical geometry in this example is if we write the edges of a graph as a matrix in the following form

$$M := \begin{bmatrix} (v_0, v_0) & (v_0, v_1) & \cdots \\ (v_1, v_0) & (v_1, v_1) & \\ \vdots & & \ddots \end{bmatrix}$$

then $(M^k)_{ij}$ returns the shortest distance from v_i to v_j after k steps.

Graph theory is a big field, but we are interested in modelling neural networks. In particular those neural networks that have specific input layers and output layers. Our goal is to get an insight of how to construct a neural network and define weights, so that our output stands in a predefined relation to our input. In particular one of the most important neural networks is the L-layer feedforward neural network. We will define and motivate L-layer feedforward neural networks using graphs at first and then abstract again to only their necessary features.

Definition 2.5. An **L-layer feedforward neural network** in graph form (G, σ) is a weighted graph $G = (V, E)$ with a piecewise differentiable activation function $\sigma : \mathbb{R} \rightarrow \mathbb{R}$. The graph consists of L sets $V^{(j)}$, $j = 1, \dots, L$ of vertices $V = \cup_{j=1}^L V^{(j)}$ with $L - 1$ corresponding sets of edges $E^{(i)} \subset \{(x, y) | x \in V^{(i)}; y \in V^{(i+1)}\}$, $i = 1, \dots, L - 1$ and $E = \cup_{i=1}^{L-1} E^{(i)}$ which connect two consecutive layers.

The Graph of a 0-layer feedforward neural network is an empty graph and of a 1-layer feedforward neural network is a set of vertices without connecting edges. The **activation function** σ will be applied in between propagating nodes in forward propagation 2.10.

Definition 2.6. The first layer of an L -layer feedforward neural network is called the **input layer** and the last (L -th) layer is called the **output layer**. All layers in between collectively are called **hidden layers**.

Definition 2.7. A L -layer feedforward neural network (G, σ) with $G = (V, E)$ is called **fully connected** if E is the largest out of every possible sets of E .

In a fully connected feedforward neural network every vertex of $V^{(j)}$ is connected through an edge to a vertex of $V^{(j+1)}$ for each $j = 1, \dots, L - 1$.

Example 2.8. We will construct a simple fully connected 4-layer feedforward neural network for the sake of visualisation. Using the same notation as in definition 2.5 we set $V^{(1)} = \{v_{11}, v_{12}, v_{13}, v_{14}\}$, $V^{(2)} = \{v_{21}, v_{22}, v_{23}\}$, $V^{(3)} = \{v_{31}, v_{32}, v_{33}\}$ and $V^{(4)} = \{v_{41}\}$. The edges are immediately defined when the vertices are defined in a fully connected graph. For example $E^{(3)} = \{(v_{31}, v_{41}), (v_{32}, v_{41}), (v_{33}, v_{41})\}$ is forced. As the use of weights is not introduced yet, we set them all to 1 for example and depict the graph in figure 5:

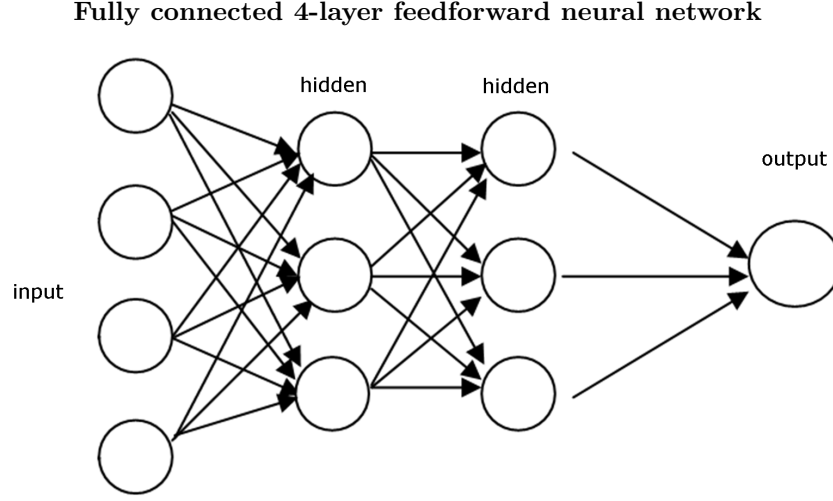


Figure 5: An example image of a fully connected feedforward neural network without visualised edge and vertex weights [GC].

Inspired by biological neural networks, we have abstracted some of the essential features, given an introduction to graphs and used these as a mathematical buildingblock for neural networks. With the introduction of feedforward neural networks we have cast a sensible form of neural networks with which we will be able to store and also learn complex relations between input data and output data through forward and backward propagation.

Forward propagation describes the process by which from an input vector the output vector is calculated. First we need to introduce the bias vectors and weight matrices.

Remark 2.9. Let (G, σ) be a L -layer feedforward neural network, with $G = (V, E, \psi, \omega)$, where we name the elements from the sets $V^{(j)} = \{b_{j1}, \dots, b_{j|V^{(j)}|}\}$ so that these are ordered. The **bias vector** of layer j :

$$b^{(j)} = (\psi(b_{j1}), \dots, \psi(b_{j|V^{(j)}|}))^t,$$

corresponds to the weights of vertices in $V^{(j)}$ for $j = 2, \dots, L$. The **weights matrix** $W^{(j)}$ with the entries

$$W_{nm}^{(j)} = \omega((b_{j-1n}, b_{jm}))$$

corresponds to the matrix of weights of edges connecting layer $j - 1$ and j where j is still in range $j = 2, \dots, L$.


We will label feedforward neural networks this way until we define feedforward neural networks as functions and can define forward propagation.

Definition 2.10. (Forward Propagation) Let (G, σ) be as in remark 2.9 a L -layer feedforward neural network. Let $a^{(1)} = x \in \mathbb{R}^{|V^{(1)}|}$ be the input vector of the same dimension as the cardinality of the first layer $V^{(1)}$ of the neural network, then recursively we define $z^{(j)} = W^{(j)}a^{(j-1)} + b^{(j)}$ and $a^{(j)} = \sigma(z^{(j)})$, for $j = 2, \dots, L$. A **forward propagation** of our L -layer feedforward neural network (G, σ) is defined to be the value of $a^{(L)}$.

To prove that the feedforward neural networks we have defined are able to be set up, so that for any specific binary input any binary specific output can be obtained, we will view the infrastructure of the weights, bias and the function σ together as a circuit. Often binary inputs, as for example black and white pictures, are fed into neural networks and binary outputs, like 0 if it depicts a dog or 1 if it depicts a cat, are expected.


Remark 2.11. Binary circuits are defined by basic logical gates like AND, OR, and NOT gates and a lot more, that manipulate binary input as shown in figure 6. A set of gates is called a **Universal Logic Gates Set** if any other logic or boolean function can be created with those gates. After [Qui55] AND, OR and NOT define such a set. This means, if we can reproduce these three gates in form of feedforward neural networks, any boolean logic can be created by these. For that we set $\omega = \mathbf{1}$, the weights of edges to one and

YES




INPUT		OUTPUT
A	B	
0	0	0
0	1	0
1	0	0
1	1	1

NOT




INPUT	OUTPUT
A	
0	1
1	0

AND




INPUT			OUTPUT
A	B	C	
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

OR




INPUT			OUTPUT
A	B	C	
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

XOR




INPUT			OUTPUT
A	B	C	
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

NAND




INPUT			OUTPUT
A	B		
0	0	1	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

NOR



INPUT			OUTPUT
A	B		
0	0	1	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

XNOR



INPUT			OUTPUT
A	B		
0	0	1	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Figure 6: Boolean Gates Truth Tables [Abe]

can reproduce these three gates in form of feedforward neural networks, any boolean logic can be created by these. For that we set $\omega = \mathbf{1}$, the weights of edges to one and

$$\sigma(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x \leq 0. \end{cases}$$

Usually the output of a neural network is non binary, but is trained so that the higher the output value is, the higher the probability of the specific outcome and the lower the output, the lower the probability, with zero being neutral. To fit this interpretation σ is defined to immediately produce a binary output. Now we will define three different 2-layer feedforward neural networks with this sigma.

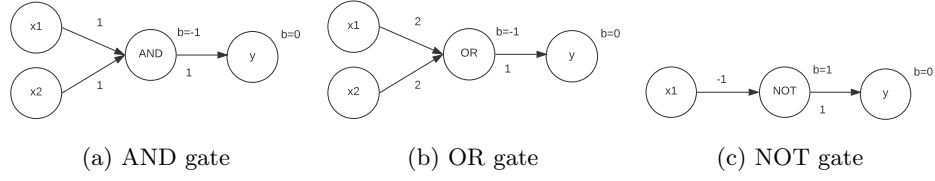


Figure 7: Neural Logical Gates

Each network takes a real vector x as input and returns y . The weights of each layer are attached to the edges and the biases are labeled with "b". The first gate, shown in figure 7 (a), defines the AND gate. It is defined with the graph $G = (E, V)$, $V = \{b_{11}, b_{12}, b_{21}, b_{31}\}$, $E = \{(b_{11}, b_{21}), (b_{12}, b_{21}), (b_{21}, b_{31})\}$ being a fully connected neural network with weights

$$\psi(b_{11}) = \psi(b_{12}) = \psi(b_{31}) = 0, \psi(b_{21}) = -1 \text{ and } W^{(1)} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, W^{(2)} = (1).$$

The first index of every bias corresponds to the layer. Lets compare the boolean output with the truth table. Inputs $\{(x, y) \mid x, y \in \{0, 1\}\}$ with

$$z^{(1)} = (1, 1) \begin{pmatrix} x \\ y \end{pmatrix} - 1 \rightarrow z^{(2)} = (1)(\sigma(x + y - 1)) + 0 = \sigma(x + y - 1)$$

$$\text{for } (0, 0) \text{ yield } z^{(2)} = \sigma(-1) = 0,$$

$$\text{for } (1, 0), (0, 1) \text{ yield } z^{(2)} = \sigma(0) = 0 \text{ and}$$

$$\text{for } (1, 1) \text{ yield } z^{(2)} = \sigma(1) = 1.$$

The same graph with weights

$$\psi(b_{11}) = \psi(b_{12}) = \psi(b_{31}) = 0, \psi(b_{21}) = -1 \text{ and } W^{(1)} = \begin{pmatrix} 2 \\ 2 \end{pmatrix}, W^{(2)} = (1)$$

makes an OR gate:

$$z^{(1)} = (2, 2) \begin{pmatrix} x \\ y \end{pmatrix} - 1 \rightarrow z^{(2)} = (1)(\sigma(2x + 2y - 1)) + 0 = \sigma(2x + 2y - 1)$$

$$\text{for } (0, 0) \text{ yield } z^{(2)} = \sigma(-1) = 0,$$

$$\text{for } (1, 0), (0, 1) \text{ yield } z^{(2)} = \sigma(2) = 1 \text{ and}$$

$$\text{for } (1, 1) \text{ yield } z^{(2)} = \sigma(4) = 1.$$

The graph of the NOT gate is different in that it only has one input. Therefore the graph also changes to $G = (E, V)$, $V = \{b_{11}, b_{21}, b_{31}\}$, $E = \{(b_{11}, b_{21}), (b_{21}, b_{31})\}$ also a fully connected 2-layer feedforward neural network. With weights

$$\psi(b_{11}) = \psi(b_{31}) = 0, \psi(b_{21}) = 1 \text{ and } W^{(1)} = (-1), W^{(2)} = (1)$$

that define a NOT gate:

$$\begin{aligned} \text{for } (0) \text{ yield } z^{(2)} &= z^{(1)} = 1 \text{ and} \\ \text{for } (1) \text{ yield } z^{(2)} &= z^{(1)} = 0. \end{aligned}$$

We have proven that any binary input can create any binary output. This also means that by scaling the input and output weights it is possible to create any specific output from any specific input.

The last abstraction level of feedforward neural networks is to define them as a function, which enables us to compare feedforward neural networks to tropical polynomials in section 4.

Definition 2.12. [ZNL18] Viewed abstractly, an **L-layer feedforward neural network** is a map $\nu : \mathbb{R}^d \rightarrow \mathbb{R}^p$ given by a composition of functions

$$\nu := \sigma^{(L)} \circ \rho^{(L)} \circ \sigma^{(L-1)} \circ \rho^{(L-1)} \circ \dots \circ \sigma^{(1)} \circ \rho^{(1)}.$$

The preactivation functions $\rho^{(1)}, \dots, \rho^{(L)}$ are affine transformations to be determined and the activation functions $\sigma^{(1)}, \dots, \sigma^{(L)}$ are chosen and fixed in advance.

We denote the width, i.e. the number of nodes, of the l -th layer by $n_l, l = 1, \dots, L - 1$. We set $n_0 := d$ and $n_L := p$, respectively the dimensions of the input and output of the network. The output from the l -th layer will be denoted by

$$\nu^{(l)} = \sigma^{(l)} \circ \rho^{(l)} \circ \sigma^{(l-1)} \circ \rho^{(l-1)} \circ \dots \circ \sigma^{(1)} \circ \rho^{(1)},$$

i.e. it is a map $\nu^{(l)} : \mathbb{R}^d \rightarrow \mathbb{R}^{n_l}$. For convenience, we assume $\nu^{(0)}(x) := x$.

The affine function $\nu^{(l)} : \mathbb{R}^{n_{l-1}} \rightarrow \mathbb{R}^{n_l}$ is given by a weight matrix $A^{(l)} \in \mathbb{Z}^{n_l \times n_{l-1}}$ and a bias vector $b^{(l)} \in \mathbb{R}^{n_l}$:

$$\rho^{(l)}(\nu^{(l-1)}) := A^{(l)}\nu^{(l-1)} + b^{(l)}.$$

The (i, j) -th coordinate of $A^{(l)}$ will be denoted $a_{ij}^{(l)}$ and the i -th coordinate of $b^{(l)}$ by $b_i^{(l)}$. Collectively they form the parameters of the l -th layer.

This way forward propagation corresponds to evaluation of the neural network.

Remark 2.13. [ZNL18] For a vector input $x \in \mathbb{R}^{n_l}$, $\sigma^{(l)}(x)$ is understood to be in coordinatewise sense; so $\sigma : \mathbb{R}^{n_l} \rightarrow \mathbb{R}^{n_l}$.

Remark 2.14. [ZNL18] We assume the final output of a neural network $\nu(x)$ is fed into a **score function** $s : \mathbb{R}^p \rightarrow \mathbb{R}^m$ that is application specific. The score function is quite often regarded as the last layer of a neural network but this is purely a matter of convenience and we will not assume this.

The next part of the section will be devoted to giving a brief introduction to back-propagation of neural networks. It is key to the understanding of modern applications of feedforward neural networks. The underlying analytics for the small section of back-propagation will not be carried out in detail, rather it will be required as prerequisite knowledge from the reader.

Backpropagation is best explained by viewing a feedforward neural network as a function or a concatenation of functions, as in definition 2.12, which we can extend to a cost function and perform gradient descent on. In the following we may refer to $A^{(l)}$ as $w^{(l)}$ the **weight matrices** and $b^{(l)}$ simply as the **bias vectors**.

Remark 2.15. For a multi-variate function $F(X)$ that is defined and differentiable in a neighbourhood of a point a , then F increases in point a fastest in value in the direction of the gradient $\text{grad}(F)(a)$.

This motivates the following definition.

Definition 2.16. (Gradient descent) It follows from remark 2 that for a differentiable function $F : \mathbb{R} \rightarrow \mathbb{R}$ if

$$a_{n+1} = a_n - \epsilon \cdot \text{grad}(F)(a_n)$$

for $\epsilon \in \mathbb{R}_+$ small enough and $a_n \in \mathbb{R}$ that $F(a_n) \geq F(a_{n+1})$. With this observation in mind, one starts with a guess $x_0 \in \mathbb{R}$ for a local minimum of F , defines the sequence $(x_n)_{n \in \mathbb{N}}$ as above ($c_{n+1} = c_n - \epsilon \cdot \text{grad}(F)(x_n)$) and with epsilon small enough x_n will eventually end up close to a local minimum. Setting up and calculating the sequence x_n to step towards local minima is called gradient descent.

We will use gradient descent with a cost function that we want to optimize the neural network on. To optimize a feedforward neural network to give sensible outputs relative to the inputs, we define the cost function.

Definition 2.17. (Cost function) Let $\nu : \mathbb{R}^d \rightarrow \mathbb{R}^p$ be a L -layer feedforward neural network and $s : \mathbb{R}^p \rightarrow \mathbb{R}^m$ a score function. The cost function for ν to a specific input x and desired output y is C_x or $C_x(s(\nu(x)), y)$ being a differentiable extension of ν dependant on the weights and biases of ν . We will assume the score function is considered in the definition of the cost function and simply write $C_x(\nu(x), y)$ instead. Notice that the input-output pair (x, y) is fixed, where the weights $w^{(l)}$ and biases $b^{(l)}$ are variable. The cost function of a finite set \mathcal{X} of input vectors is then the average

$$C = \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} C_x$$

over cost functions C_x for individual training examples, x .

Backpropagation is an algorithm to tweak weights of a neural network to minimize the cost function on a training set, containing input vectors and the corresponding desired outputs. For our theoretical analysis of backpropagation we limit ourselves to the case where the σ of feedforward neural networks are all differentiable. For a discussion of the general case of piecewise linear activation functions, we refer the reader to [AHSB14].

Definition 2.18. (Backpropagation) Let ν be an L -layer feedforward neural network, $\mu \in \mathbb{R}$ a learning rate and C be a cost function over ν as in 2.17. Application of backpropagation to ν is calculating $\text{grad}_{w^{(i)}}(C)$ the gradient of the weights and $\text{grad}_{b^{(i)}}(C)$ the gradient of the biases of the cost function for $i = 1, \dots, L$ and applying changes to the weights and biases of the neural network ν as follows:

$$\begin{aligned} w^{(i)} &\rightarrow (w^{(i)})' = w^{(i)} - \mu \cdot \text{grad}_{w^{(i)}}(C) \\ b^{(i)} &\rightarrow (b^{(i)})' = b^{(i)} - \mu \cdot \text{grad}_{b^{(i)}}(C) \text{ for } i = 1, \dots, L. \end{aligned}$$

Repeated application of backpropagation is a gradient descent algorithm and therefore optimizes the cost function and approaches local minima. This gives a way to train a neural network on a datapool efficiently.

Remark 2.19. In essence, backpropagation evaluates the expression for the derivative of the cost function as a product of derivatives between each layer. This happens from left to right in a "backward" manner, where the gradient of σ 's between each layer is a simple modification of the partial products (the "backwards propagated error") [Nie17]. For an input-output pair (x, y) , the loss is:

$$C_x(\sigma^{(L)} \circ \rho^{(L)} \circ \sigma^{(L-1)} \circ \rho^{(L-1)} \circ \dots \circ \sigma^{(1)} \circ \rho^{(1)}(x), y).$$

The gradient grad is the transpose of the derivative of the output in terms of the input. To understand how the algorithm of gradient descent operates, we first interpret the constant x as a variable and $b^{(l)}, w^{(l)}$ as constants again. Then with the chain rule the the gradient in x is the following product:

$$\begin{aligned} \frac{\partial C_x}{\partial x} &= \frac{\partial C_x}{\partial \sigma^{(L)}} \cdot \frac{\partial \sigma^{(L)}}{\partial \rho^{(L)}} \cdot \frac{\partial \rho^{(L)}}{\partial \sigma^{(L-1)}} \cdot \dots \cdot \frac{d\sigma^{(1)}}{d\rho^{(1)}} \cdot \frac{\partial \rho^{(1)}}{\partial x} \\ &= C'_x(\overline{\sigma^{(L)}}(x)) \cdot (\sigma^{(L)})'(\overline{\rho^{(L)}}(x)) \cdot \dots \cdot (\sigma^{(1)})'(\overline{\rho^{(1)}}(x)) \cdot (\rho^{(1)})'(x) \\ \text{grad}_x(C_x) &= \left(\frac{\partial C}{\partial x} \right)^t. \end{aligned}$$

Where $\overline{\sigma^{(l)}}$ and $\overline{\rho^{(l)}}$ are defined as:

$$\begin{aligned} \overline{\sigma^{(l)}} &= \sigma^{(l)} \circ \rho^{(l)} \circ \dots \circ \sigma^{(1)} \circ \rho^{(1)} \text{ and} \\ \overline{\rho^{(l)}} &= \rho^{(l)} \circ \sigma^{(l-1)} \circ \dots \circ \sigma^{(1)} \circ \rho^{(1)}. \end{aligned}$$

We can use the property that the derivative of the cost function is the product of derivatives between each layer and the fact, that $\overline{\sigma^{(l)}}$ and $\overline{\rho^{(l)}}$ get calculated and may be saved in a forward propagation, which is needed to calculate x before backpropagating. The affine linear functions $\rho^{(l)}$ for $l = L, \dots, 1$ can be written unambiguously as it's weight matrix $w^{(l)}$ and bias vector $b^{(l)}$, $\rho^{(l)} = x^{(l)}x + b^{(l)}$. With this in mind we set x constant again and in the l -th step view $w^{(l)}$ and $b^{(l)}$ as variable again. Then we set

$$\begin{aligned} \delta^{(L)} &:= \frac{\partial C}{\partial \sigma^{(L)}} \cdot \frac{\partial \sigma^{(L)}}{\partial \rho^{(L)}} \\ \delta^{(l-1)} &:= \delta^{(l)} \frac{\partial \rho^{(l)}}{\partial \sigma^{(l-1)}} \cdot \frac{\partial \sigma^{(l-1)}}{\partial \rho^{(l-1)}}, \end{aligned}$$

for $l = L - 1, \dots, 1$. This, depending on the choice of sigma, gives an easy to calculate algorithm for the gradients:

$$\begin{aligned} \text{grad}_{w^{(l)}}(C_x) &= \left[\delta^{(l)} \cdot \frac{\partial \rho^{(l)}}{\partial w^{(l)}} \right]^T \\ \text{grad}_{b^{(l)}}(C_x) &= \left[\delta^{(l)} \cdot \frac{\partial \rho^{(l)}}{\partial b^{(l)}} \right]^T. \end{aligned}$$

This is much faster than calculating the deltas in a forward manner which would have the following form:

$$\begin{aligned}\delta^{(1)} &= \frac{\partial C_x}{\partial \sigma^{(L)}} \cdot \frac{\partial \sigma^{(L)}}{\partial \rho^{(L)}} \cdot \frac{\partial \rho^{(L)}}{\partial \sigma^{(L-1)}} \cdot \dots \cdot \frac{d\sigma^{(1)}}{\partial \rho^{(1)}} \\ &\vdots \\ \delta_w^{(L)} &= \frac{\partial C}{\partial \sigma^{(L)}} \cdot \frac{\partial \sigma^{(L)}}{\partial \rho^{(L)}}.\end{aligned}$$

The recursive way of propagating the neural network backwards is where backpropagation gets its name from. The $\sigma^{(l)}$ and $\rho^{(l)}$ for $l = 1, \dots, L$ get computed and saved beforehand in a forward propagation run to save computational time.

Now if we have a training set \mathcal{X} of input and desired output pairs, what we want to compute is

$$\begin{aligned}\text{grad}_{w^{(l)}}(C) &= \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \text{grad}_{w^{(l)}}(C_x) \\ \text{grad}_{b^{(l)}}(C) &= \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \text{grad}_{b^{(l)}}(C_x)\end{aligned}$$

the gradients corresponding to the set. Because gradient decent is based on iterative application, usually \mathcal{X} is split up into *batches*, these are subsets of \mathcal{X} of a set cardinality. This way the weights and biases get optimized on the batches iteratively. This method gets especially effective on large datapools \mathcal{X} .

This gives a way to train neural networks on a datapool. There are a number of further topics to understand state of the art training of neural networks, for example stochastic gradient descent or over fitting [Bis06].

3 Tropical algebra of neural networks

This section proves the key theorems of the thesis, an equivalence of neural networks and tropical rational functions, which allows studies on tropical rational functions in return to enforce statements on neural networks. For a tropical characterisation of neural networks we will need to restrict feedforward neural networks.

Convention 3.1. [ZNL18] We will make the following mild assumptions on the architecture of our L -layer feedforward neural networks:

- (a) the weight matrices $A^{(1)}, \dots, A^{(L)}$ are integer-valued;
- (b) the bias vectors $b^{(1)}, \dots, b^{(L)}$ are real-valued;
- (c) the activation functions $\sigma^{(1)}, \dots, \sigma^{(L)}$ take the form

$$\sigma^{(l)}(x) := \max\{x, t^{(l)}\}$$

where $t^{(l)} \in (\mathbb{R} \cup \{-\infty\})^{n_l}$ is called a threshold vector.

Since x is a vector here and therefore may be multidimensional. The activation function, here \max , is meant to be applied element wise:

$$\max\{x, t^{(l)}\} = \begin{pmatrix} \max\{x_1, t_1^{(l)}\} \\ \vdots \\ \max\{x_{n_l}, t_{n_l}^{(l)}\} \end{pmatrix}.$$

Convention 3.2. [ZNL18] The assumption (b) is completely general but there is also no loss of generality in (a), i.e., in restricting the weights A^1, \dots, A^L from real matrices to integer matrices, as:

- real weights can be approximated arbitrarily closely by rational weights;
- one may then ‘clear denominators’ in these rational weights by multiplying them by the least common multiple of their denominators to obtain integer weights;
- keeping in mind that scaling all weights and biases by the same positive constant has no bearing on the workings of a neural network.

Henceforth all neural networks in our subsequent discussions will be assumed to satisfy (a)–(c).

We identify feedforward neural networks with tropical polynomial maps. This is a core insight of the paper.

Theorem 3.3. [ZNL18] (Tropical characterization of neural networks).

A feedforward neural network that fulfils the assumptions (a)–(c) is a function $\nu : \mathbb{R}^d \rightarrow \mathbb{R}^p$ whose coordinates are tropical rational functions of the input, i.e.,

$$\nu(x) = F(x) \odot G(x) = F(x) - G(x)$$

where F and G are tropical polynomial maps. Thus ν is a tropical rational map.

Proof. Consider the output from the first layer in a neural network

$$\nu(x) = \max\{Ax + b, t\};$$

where $A \in \mathbb{Z}^{p \times d}$, $b \in \mathbb{R}^p$, and $t \in (\mathbb{R} \cup \{-\infty\}^p)$. We will decompose A as a difference of two nonnegative integervalued matrices, $A = A_+ - A_-$ with $A_+, A_- \in \mathbb{N}^{p \times d}$; e.g., in the standard way with entries

$$a_{ij}^+ := \max\{a_{ij}, 0\}, \quad a_{ij}^- := \max\{-a_{ij}, 0\}$$

respectively. Since

$$\max\{Ax + b, t\} = \max\{A_+x + b, A_-x + t\} - A_-x$$

and

$$\begin{aligned} & \max\{A_+x + b, A_-x + t\} - A_-x \\ &= \begin{pmatrix} \max\{a_{1,1}x_1 + \cdots + a_{1d}x_d + b_1\} - \max\{t_1, -\infty\} \\ \vdots \\ \max\{a_{p,1}x_1 + \cdots + a_{pd}x_d + b_p\} - \max\{t_p, -\infty\} \end{pmatrix}, \end{aligned}$$

we see that every coordinate of one-layer neural network is a difference of two tropical polynomials, which means the first layer of a neural network can be described through a tropical rational map.

Now let $A^{(l+1)} \in \mathbb{Z}^{m \times n}$, $b^{(l+1)} \in \mathbb{R}^m$ be the parameters of the $(l+1)$ -th layer, and let $t \in (\mathbb{R} \cup -\infty)^m$ be the threshold vector in the $(l+1)$ -th layer. If the nodes of the l -th layer are given by a tropical rational map

$$\nu^{(l)}(x) = F^{(l)}(x) \odot G^{(l)}(x) = F^{(l)}(x) - G^{(l)}(x),$$

i.e., each coordinate of $F^{(l)}$ and $G^{(l)}$ is a tropical polynomial in x . We want to show, that then $\nu^{(l+1)}$ is also a rational map. We begin by calculating

$$\begin{aligned} \rho^{(l+1)} \circ \nu^{(l)}(x) &= A^{(l+1)}\nu^{(l)} + b^{(l+1)} \\ &= A^{(l+1)}(F^{(l)}(x) - G^{(l)}(x)) + b^{(l+1)} \\ &= (A_+^{(l+1)} - A_-^{(l+1)})(F^{(l)}(x) - G^{(l)}(x)) + b^{(l+1)} \\ &= A_+^{(l+1)}F^{(l)}(x) + A_-^{(l+1)}G^{(l)}(x) + b^{(l+1)} \\ &\quad - (A_+^{(l+1)}G^{(l)}(x) + A_-^{(l+1)}F^{(l)}(x)) \\ &=: H^{(l+1)}(x) - G^{(l+1)}(x), \end{aligned}$$

with

$$\begin{aligned} H^{(l+1)}(x) &:= A_+^{(l+1)}F^{(l)}(x) + A_-^{(l+1)}G^{(l)}(x) + b^{(l+1)}, \\ G^{(l+1)}(x) &:= A_+^{(l+1)}G^{(l)}(x) + A_-^{(l+1)}F^{(l)}(x). \end{aligned}$$

is a rational map since when we write $g_i^{(l)}$ and $h_i^{(l)}$ for the i -th coordinate of $G^{(l)}$ and $H^{(l)}$ respectively. In tropical arithmetic, the recurrence above takes the form

$$\begin{aligned} g_i^{(l+1)} &= [\odot_{j=1}^n (g_j^{(l)})^{a_{ij}^+}] \odot [\odot_{j=1}^n (f_j^{(l)})^{a_{ij}^-}] \in \mathbb{T}(x), \\ h_i^{(l+1)} &= [\odot_{j=1}^n (f_j^{(l)})^{a_{ij}^+}] \odot [\odot_{j=1}^n (g_j^{(l)})^{a_{ij}^-}] \odot b_i \in \mathbb{T}(x). \end{aligned}$$

On this basis it is easy to show that ν

$$\begin{aligned} \nu^{(l+1)}(x) &= \sigma^{(l+1)} \circ \rho^{(l+1)} \circ \nu^{(l)}(x) \\ &= \sigma^{(l+1)}(H^{(l+1)}(x) - G^{(l+1)}(x)) \\ &= \max\{H^{(l+1)}(x) - G^{(l+1)}(x), t\} \\ &= \max\{H^{(l+1)}(x), t + G^{(l+1)}(x)\} - G^{(l+1)}(x) \\ &=: F^{(l+1)}(x) - G^{(l+1)}(x), \end{aligned}$$

with

$$F^{(l+1)}(x) := \max\{H^{(l+1)}(x), t + G^{(l+1)}(x)\}$$

is a tropical rational map considering with f_i denoting the i th coordinate of $F^{(l)}$ and

$$f_i^{(l+1)} = h_i^{(l+1)} \oplus (g_i^{(l+1)} \odot t_i) \in \mathbb{T}(x),$$

$f_i^{(l+1)}$ is a tropical rational function. Through induction this ends the proof. \square

Remark 3.4. [ZNL18] Note that the tropical rational functions G and H in theorem 3.3 have real coefficients, not integer coefficients. The integer weights $A^{(l)} \in \mathbb{Z}^{n_l \times n_{l-1}}$ have gone into the powers of tropical monomials in f and g , which is why we require our weights to be integer-valued, although, as we have explained, this requirement imposes little loss of generality.

Definition 3.5. Let $\nu : \mathbb{R}^d \rightarrow \mathbb{R}^p$ be an L -layer neural network with

$$\nu = \sigma^{(L)} \circ \rho^{(L)} \circ \sigma^{(L-1)} \circ \rho^{(L-1)} \circ \dots \circ \sigma^{(1)} \circ \rho^{(1)}.$$

Then ν has **ReLU activation** if $\sigma^{(1)} = \dots = \sigma^{(L-1)} = \max\{x, 0\}$ and $\sigma^{(L)} = \max\{x, -\infty\}$.

The result of the following corollary is immediate.

Corollary 3.6. [ZNL18] Let $\nu : \mathbb{R}^d \rightarrow \mathbb{R}$ be an ReLU activated feedforward neural network with integer weights and linear output. Then ν is a tropical rational function.

A more remarkable fact is the converse of Corollary 3.6.

Theorem 3.7. [ZNL18] (Equivalence of neural networks and tropical rational functions).

- (i) Let $\nu : \mathbb{R}^d \rightarrow \mathbb{R}$. Then ν is a tropical rational function if and only if ν is a feedforward neural network satisfying assumptions (a)–(c).
- (ii) A tropical rational function $f \oslash g : \mathbb{R}^d \rightarrow \mathbb{R}$ can be represented as an L -layer neural network, with

$$L \leq \max\{\lceil \log_2 r_f \rceil, \lceil \log_2 r_g \rceil\} + 2,$$

where r_f and r_g are the number of monomials in the tropical polynomials f and g respectively.

Proof. That a feedforward neural network under assumptions (a)-(c) is a tropical polynomial map has been shown in theorem 3.3. Now if the neural network has ReLU activations, we want to show the opposite implication, as in theorem 3.3, we will show this by induction. Let $\nu = c_1 x^{\alpha_1} \oplus \dots \oplus c_r x^{\alpha_r}$ be a tropical polynomial as in definition 1.8 with $\nu : \mathbb{R}^d \rightarrow \mathbb{R}$. Setting $\sigma_t := \max\{x, t\}$, our base case is $\nu(x) = cx^\alpha$ a tropical monomial. Considering

$$\nu(x) = cx^\alpha = \max\{\alpha^t x + c, -\infty\} = (\sigma_{-\infty} \circ \rho)(x),$$

where \max is applied coordinatewise and $\rho(x) = a^t x + b$ is a real affine linear function, ν is a feedforward neural network under assumptions (a)-(c). As induction step, we observe two tropical polynomials p and q which are represented as neural networks with l_p and l_q layers respectively,

$$\begin{aligned} p(x) &= (\sigma_{-\infty} \circ \rho_p^{(l_p)} \circ \sigma_0 \circ \dots \circ \sigma_0 \circ \rho_p^{(1)})(x), \\ q(x) &= (\sigma_{-\infty} \circ \rho_q^{(l_q)} \circ \sigma_0 \circ \dots \circ \sigma_0 \circ \rho_q^{(1)})(x). \end{aligned}$$

Here all sigma are set to σ_0 except the last one to $\sigma_{-\infty}$ as prerequisite by the ReLU activation. We will write $p \oplus q$ as a neural network:

$$\begin{aligned} (p \oplus q)(x) &= \min\{p(x), q(x)\} \\ &= \sigma_{-\infty}(\min\{p(x), q(x)\}) \\ &= \sigma_{-\infty}(\min(p(x) - q(x), 0) + \min(q(x), 0) - \min(-q(x), 0)) \\ &= \sigma_{-\infty} \left(\begin{pmatrix} 1 & 1 & -1 \end{pmatrix} (\sigma_0(\rho(y(x))) + \begin{pmatrix} 0 & 0 & 0 \end{pmatrix}^T) \right) \end{aligned}$$

where $y : \mathbb{R}^d \rightarrow \mathbb{R}^2$ is given by $y(x) = (p(x), q(x))$ and $\rho_i : \mathbb{R}^2 \rightarrow \mathbb{R}$, $i = 1, 2, 3$, are linear functions defined by

$$\rho_1(y) = y_1 - y_2, \rho_2(y) = y_2, \rho_3(y) = -y_2.$$

The function $y(x)$ is a $\max\{l_p, l_q\}$ layer feedforward neural network. This is easily seen by extending the shorter neural network p or q with identity ρ 's, say w.l.o.g. $l_p < l_q$, then $\rho_p^{(i)} = id$ for $i = l_p + 1, \dots, l_q$. This way $y(x) = (\sigma_{-\infty} \circ \rho^{(l_p)} \circ \sigma_0 \circ \dots \circ \sigma_0 \circ \rho^{(1)})(x)$ with $\rho^{(i)}(x) = (\rho_p^{(i)}(x), \rho_q^{(i)}(x))$. By removing the last irrelevant sigma $\sigma_{-\infty}$ off of p and q and since composition of affine linear functions are affine linear, $(p \oplus q)$ is a neural network with $\max\{l_p, l_q\} + 1$ layers. Thus, by induction, any tropical polynomial can be written as a neural network with ReLU activation.

Observe also that if a tropical polynomial is the tropical sum of r monomials, then it can be written as a neural network with no more than $\lceil \log_2 r \rceil + 1$ layers. Once more we write $p \odot q$, where p and q are tropical polynomials as a neural network. Under the same assumptions as above and with

$$\begin{aligned} (p \odot q)(x) &= \sigma_0(p(x)) - \sigma_0(-p(x)) + \sigma_0(-q(x)) - \sigma_0(q(x)) \\ &= \sigma_{-\infty}(\begin{pmatrix} 1 & -1 & 1 & -1 \end{pmatrix} \sigma_0(\rho(y(x)))) \end{aligned}$$

where $\rho_i : \mathbb{R}^2 \rightarrow \mathbb{R}^1$, $i = 4, 5, 6, 7$, are linear functions defined by

$$\rho_4(y) = y_1, \rho_5(y) = -y_1, \rho_6(y) = -y_2, \rho_7(y) = y_2.$$

The same argumentation as above shows, $p \odot q$ is also a neural network with at most $\max\{l_p, l_q\}$ layers.

Finally, if f and g are tropical polynomials that are respectively tropical sums of r_f and r_g monomials, then the discussions above show that $(f \odot g) = f(x) - g(x)$ is a neural network with at most $\max\{\lceil \log_2 r_f \rceil + 1, \lceil \log_2 r_g \rceil + 1\} + 1 = \max\{\lceil \log_2 r_f \rceil, \lceil \log_2 r_g \rceil\} + 2$ layers. \square

Proposition 3.8. [ZNL18] Let $\nu : \mathbb{R}^d \rightarrow \mathbb{R}$. Then ν is a continuous piecewise linear function with integer coefficients if and only if ν is a tropical rational function.

Proof. We have shown in lemma 1.13 that $\nu \in \mathbb{T}(X_1, \dots, X_d)$ with $\nu : \mathbb{R}^d \rightarrow \mathbb{R}$ is a piecewise linear function and has integer coefficients per definition. Therefore we only have to show the opposite implication. Let $\nu : \mathbb{R}^d \rightarrow \mathbb{R}$ be a piecewise linear function with integer coefficients. This means we can divide \mathbb{R}^d into N polyhedral regions on each of which ν restricts to a linear function

$$l_i(x) = a_i^t x + b$$

where $a_i \in \mathbb{Z}^d, b_i \in \mathbb{R}$ and $i = 1, \dots, L$. Theorem 4.1 in [TM99] shows, that we can find N subsets of $\{1, \dots, L\}$, denoted by $S_j, j = 1, \dots, N$, so that ν has a representation

$$\nu(x) = \max_{j=1, \dots, N} \min_{i \in S_j} l_i.$$

Since l_i is a linear function for $i = 1, \dots, L$, we know $l_i \in \mathbb{T}(X_1, \dots, X_d)$ is a tropical rational function. We show for two tropical rational functions $p, q \in \mathbb{T}(X_1, \dots, X_k)$ for any $k \in \mathbb{N}$, that their minimum is also a tropical rational function:

$$\min\{p, q\} = p + q - \max\{p, q\} = [p \odot q] \odot [p \oplus q] \in T(X_1, \dots, X_k).$$

Therefore by induction $\min_{i \in S_j} l_i \in \mathbb{T}(X_1, \dots, X_d)$ holds for any $j = 1, \dots, N$ and we have shown that ν is a tropical rational function. \square

Remark 3.9. [ZNL18] Corollary 3.6, theorem 3.7, and proposition 3.8 collectively imply the equivalence of

- (i) tropical rational functions,
- (ii) continuous piecewise linear functions with integer coefficients,
- (iii) neural networks satisfying assumptions (a)-(c).

We have thus introduced the key insight of this thesis, the equivalence of remark 3.9 allows for knowledge in tropical algebra to be applied to neural networks. The following sections will be centred around exactly that. In particular the goal is to find an upper bound on the number of linear regions, defined in section 4, of a real-valued feedforward neural network satisfying (a)-(c), presented in theorem 5.3.

4 Tropical hypersurfaces

In this section we will start viewing tropical polynomials as geometric figures with a tropical hypersurface that is piecewise linear. Recall that we write a tropical polynomial as $c_1x^{\alpha_1} \oplus \dots \oplus c_rx^{\alpha_r}$ where $\alpha_i = (\alpha_{i1}, \dots, \alpha_{id}) \in \mathbb{N}^d$ and $c_i \in \mathbb{T}, i = 1, \dots, r$ and also $\alpha_i \neq \alpha_j$ for any $i \neq j$. We begin by defining the tropical hypersurface of a tropical polynomial.

Definition 4.1. [ZNL18, p. 3] The **tropical hypersurface** of a tropical polynomial $f(x) = c_1x^{\alpha_1} \oplus \dots \oplus c_rx^{\alpha_r}$ is

$$\mathcal{T}(f) := \{x \in \mathbb{R}^d : c_ix^{\alpha_i} = c_jx^{\alpha_j} = f(x) \text{ for some } \alpha_i \neq \alpha_j\}$$

i.e., the set of points x at which the value of f at x is attained by two or more monomials in f .

As an example we introduce the tropical hypersurface of a quadratic polynomial.

Example 4.2. [MS15, p. 23] Suppose we consider the general quadratic polynomial

$$p(x, y) = a \odot x^2 \oplus b \odot xy \oplus c \odot y^2 \oplus d \odot y \oplus e \oplus f \odot x$$

and suppose that the coefficients $a, b, c, d, e, f \in \mathbb{R}$ satisfy the inequalities

$$2b < a + c, 2d < a + f, 2e < c + f.$$

Then the graph of $p : \mathbb{R}^2 \rightarrow \mathbb{R}$ is the lower envelope of six planes in \mathbb{R}^3 . This is shown in Figure 8, where each linear piece of the graph is labelled by the corresponding linear function. Below this "tent" lies the tropical quadratic hypersurface $\mathcal{T}(p) \subset \mathbb{R}^2$. This has four vertices, three bounded edges and six half-rays (two northern, two eastern and two southwestern).

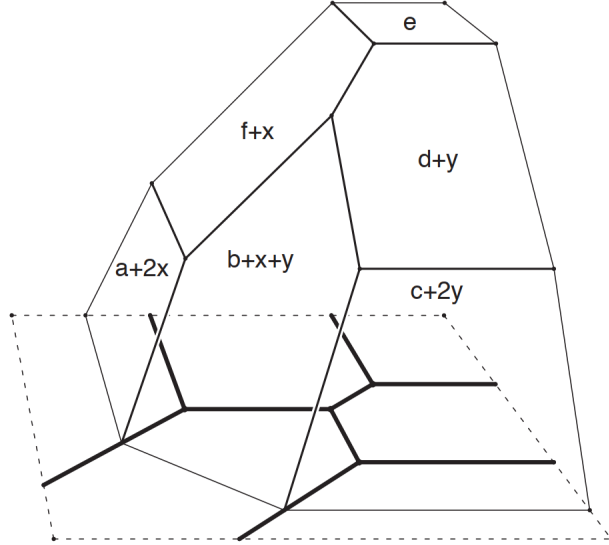


Figure 8: The graph and the curve defined by a quadratic tropical polynomial. [MS15, p. 22]

To understand the geometry of tropical rational functions, tropical polynomial maps and tropical rational maps better we will show that each are piecewise linear.

Lemma 4.3. Tropical rational functions are piecewise linear functions. Tropical polynomial maps and tropical rational maps are piecewise linear maps.

Proof. Because of lemma 1.13 tropical polynomials are piecewise linear. Tropical division, \oslash , is a linear transformation, therefore tropical rational maps are piecewise linear too. Now this means that tropical rational maps and tropical polynomial maps are also piecewise linear and the notion of linear regions applies. \square

To understand remark 4.12 we need to introduce a succession of definitions.

Definition 4.4. [ZNL18, p. 4] A **convex polyhedron** or just **polyhedron** is a set defined by linear inequalities with integer coefficients:

$$\{x \in \mathbb{R} \mid Ax \leq b\}$$

for $A \in \mathbb{Z}^{m \times d}$ and $b \in \mathbb{R}^m$. A **polygon** is a bounded polyhedron.

Definition 4.5. [ZNL18, p. 4] A **linear region** of $F \in \text{Rat}(d, m)$ is a, with respect to inclusion, maximal connected subset of the domain on which F is linear. The number of linear regions of F is denoted $\mathcal{N}(f)$.

A linear region as in 4.5 is a polyhedron.

Definition 4.6. A **face** of a polyhedron $P \subset \mathbb{R}^n$ is a subset of the form

$$F = P \cap \{x \in \mathbb{R}^n \mid c^t x = b_0\},$$

where $c \in \mathbb{R}^n$ and $b_0 \in \mathbb{R}$ are picked so that

$$P \subset \{x \in \mathbb{R}^n \mid c^t x \leq b_0\}$$

holds.

Definition 4.7. [MS15] A **polyhedral complex** is a collection of polyhedra satisfying two conditions:

- (i) if a polyhedron P is in the collection, then so is any face of P
- (ii) if P and Q lie in the collection then $P \cap Q$ is a face of both P and Q .

Definition 4.8. Let Σ be a polyhedral complex and $d \in \mathbb{N}$, then the **d-skeleton** of Σ is the conjunction of all faces of Σ of dimension $\leq d$.

Definition 4.9. [MS15] The **support** $\text{supp}(\Sigma)$ of a polyhedral complex Σ is the set $\{x \in \mathbb{R}^n \mid x \in P \text{ for some } P \in \Sigma\}$.

The linearity of a tropical polynomial per definition will be broken into its hypersurface and will be linear everywhere else. Therefore the tropical hypersurface $\mathcal{T}(f)$ of a tropical polynomial f divides the domain of f into convex cells on each of which f is linear. Tropical hypersurfaces of polynomials in two variables are called **tropical curves**. [ZNL18, p. 3]

Definition 4.10. Let $A \subset \mathbb{R}^n$ be a real subset with $n \in \mathbb{N}$. Then the set $\text{Conv}(A) = \{\lambda x + (1 - \lambda)y \mid x, y \in A \text{ and } \lambda \in [0, 1] \subset \mathbb{R}\}$ is called the **convex hull** of A .

Definition 4.11. [ZNL18, p. 3] The **Newton polygon** of a tropical polynomial $f(x) = c_1 x^{\alpha_1} \oplus \dots \oplus c_r x^{\alpha_r}$ is the convex hull of $\alpha_1, \dots, \alpha_r \in \mathbb{N}^d$, regarded as points in \mathbb{R}^d ,

$$\Delta(f) := \text{Conv}\{\alpha_i \in \mathbb{R}^d : c_i \neq -\infty, i = 1, \dots, r\}.$$

Remark 4.12. [ZNL18, p. 3] Let $f = c_1 x^{\alpha_1} \oplus \dots \oplus c_r x^{\alpha_r}$ be a tropical polynomial. First we lift each α_i from \mathbb{R}^d into \mathbb{R}^{d+1} by appending c_i as the last coordinate. Denote the convex hull of the lifted $\alpha_1, \dots, \alpha_r$ as

$$\mathcal{P}(f) := \text{Conv}\{(\alpha_i, c_i) \in \mathbb{R}^d \times \mathbb{R} : i = 1, \dots, r\}.$$

Next let $UF((P)(f))$ denote the collection of upper faces in $\mathcal{P}(f)$ and $\pi : \mathbb{R}^d \times \mathbb{R} \rightarrow \mathbb{R}^d$ be the projection that drops the last coordinate. The dual subdivision determined by f is then

$$\delta(f) := \{\pi(p) \subset \mathbb{R}^d : p \in UF(\mathcal{P}(f))\}.$$

$\delta(f)$ forms a polyhedral complex with support $\delta(f)$. By [MS15, Proposition 3.1.6.], the tropical hypersurface $\mathcal{T}(f)$ is the $(d - 1)$ -skeleton of the polyhedral complex dual to $\delta(f)$. This means that each vertex in $\delta(f)$ corresponds to one "cell" in \mathbb{R}^d where the function f is linear. Thus, the number of vertices in $\mathcal{P}(f)$ provides an upper bound on the number of linear regions of f .

Our analysis of neural networks requires figuring out how the polytope $\mathcal{P}(f)$ transforms under tropical power, sum, and product.

Proposition 4.13. [ZNL18, p. 4] Let f be a tropical polynomial and let $a \in \mathbb{N}$, then

$$\mathcal{P}(f^a) = a\mathcal{P}(f).$$

The set $a\mathcal{P}(f) = \{ax : x \in \mathcal{P}(f)\} \subset \mathbb{R}^{d+1}$ is a scaled version of $\mathcal{P}(f)$ with the same shape but different volume.

Proof. Let $f : \mathbb{T}^n \rightarrow \mathbb{T}$ with $f(x) = c_1 x^{\alpha_1} \oplus \dots \oplus c_r x^{\alpha_r}$ be a tropical polynomial, as in definition 1.8. Then with $\mathcal{P}(f) = \text{Conv}\{(\alpha_i, c_i) \in \mathbb{R}^d \times \mathbb{R}, i = 1, \dots, r \text{ and } c_i \neq -\infty\}$:

$$\begin{aligned} f^a &= (c_1 x^{\alpha_1} \oplus \dots \oplus c_r x^{\alpha_r})^a \\ &= a \cdot (c_1 x^{\alpha_1} \oplus \dots \oplus c_r x^{\alpha_r}) \\ &= a \cdot (c_1 x^{\alpha_1}) \oplus \dots \oplus a \cdot (c_r x^{\alpha_r}) \\ &= (a \cdot c_1) x^{a \cdot \alpha_1} \oplus \dots \oplus (a \cdot c_r) x^{a \cdot \alpha_r} \end{aligned}$$

and therefore

$$\mathcal{P}(f^a) = \text{Conv}\{(a \cdot \alpha_i, a \cdot c_i) \in \mathbb{R}^d \times \mathbb{R}, i = 1, \dots, r \text{ and } c_i \neq -\infty\} = a\mathcal{P}(f).$$

□

A generalisation of normal summation, now on sets, is the Minkowski sum.

Definition 4.14. [ZNL18, p. 4] The **Minkowski sum** of two sets P_1 and P_2 in \mathbb{R}^d is the set

$$P_1 + P_2 := \{x_1 + x_2 \in \mathbb{R}^d : x_1 \in P_1, x_2 \in P_2\};$$

and for $\lambda_1, \lambda_2 \geq 0$, their weighted Minkowski sum is

$$\lambda_1 P_1 + \lambda_2 P_2 := \{\lambda_1 x_1 + \lambda_2 x_2 \in \mathbb{R}^d : x_1 \in P_1, x_2 \in P_2\}.$$

One important example for a Minkowski sum is a zonotope. For that we need a notion of a line segment.

Definition 4.15. A subset $A \subset \mathbb{V}$ of a vector space \mathbb{V} is a **line segment**, if $A = \{\lambda x + \lambda(1 - y) \mid \lambda \in [0, 1] \subset \mathbb{K}\}$ with \mathbb{K} being the scalar field of \mathbb{V} .

Definition 4.16. The Minkowski sum of line segments is called a **zonotope**.

Definition 4.17. Let $A \subset \mathbb{R}^n$ be a set. A is point symmetric around $b \in A$ if for $a \in A$ follows $a + 2(b - a) = 2b - a \in A$.

For A and b as in definition 4.17 and if A is convex, then it is clear that $b \in A$ the symmetry point must lie in A . We build an intuition of zonotopes by proving the following statement:

Lemma 4.18. A zonotope over \mathbb{R}^n is point symmetric and convex.

Proof. We prove a zonotope is point symmetric and convex by induction. As base case let $P \subset \mathbb{R}^n$ be a line segment in between $x, y \in \mathbb{R}^n$ then P is obviously point symmetric around $b = 0.5(x + y) \in \mathbb{R}^n$ and convex per definition. For the induction step let $A \subset \mathbb{R}^n$ be a convex set and point symmetric around $b \in \mathbb{R}^n$ and also let $P = \{\lambda x + (1 - \lambda)y \mid \lambda \in [0, 1] \subset \mathbb{R}^n\}$ for $x, y \in \mathbb{R}^n$. Then the Minkowski sum of the two sets is

$$\begin{aligned} A + P &= \{a + p \mid a \in A \text{ and } p \in P\} \\ &= \{a + \lambda x + (1 - \lambda)y \mid a \in A \text{ and } \lambda \in [0, 1]\}. \end{aligned}$$

It follows that $a_{A+P} \in A + P$ has the form $a_{A+P} = a + \lambda x + (1 - \lambda)y$ for $a \in A$ and $\lambda \in [0, 1]$. We set $b_{A+P} = (0.5)(x + y) + b$. Then

$$\begin{aligned} a_{A+P} + 2(b_{A+P} - a_{A+P}) &= a + \lambda x + (1 - \lambda)y + 2(0.5(x + y) + b - (a + \lambda x + (1 - \lambda)y)) \\ &= 2b - a + (1 - \lambda)x + \lambda y \in A + P \end{aligned}$$

since $2b - a \in A$ and $(1 - \lambda)x + \lambda y \in P$. Therefore a zonotope is symmetric. Similarly for convexity let $p, q \in A + P$ have the shape

$$\begin{aligned} p &= a_p + \lambda_p x + (1 - \lambda_p)y \\ q &= a_q + \lambda_q x + (1 - \lambda_q)y. \end{aligned}$$

For $\mu \in [0, 1] \subset \mathbb{R}^n$ we need to compute

$$\begin{aligned} \mu p + (1 - \mu)q &= \mu(a_p + \lambda_p x + (1 - \lambda_p)y) + (1 - \mu)(a_q + \lambda_q x + (1 - \lambda_q)y) \\ &= \mu a_p + (1 - \mu)a_q + (\mu \lambda_p + (1 - \mu)\lambda_q)x + (\mu(1 - \lambda_p) + (1 - \mu)(1 - \lambda_q))y \\ &= \mu a_p + (1 - \mu)a_q + \Phi x + (1 - \Phi)y \end{aligned}$$

with $\Phi = \Phi(\mu, \lambda_p, \lambda_q) = \mu\lambda_p + \lambda_q - \mu\lambda_q$. If $0 \leq \Phi(\mu, \lambda_p, \lambda_q) \leq 1$, then a zonotope would be convex, since $(1 - \mu)a_q + \Phi x + (1 - \Phi)y \in P$ would hold and $\mu a_p + (1 - \mu)a_q \in A$ holds anyway. So to conclude we show $0 \leq \Phi(\mu, \lambda_p, \lambda_q) \leq 1$. Because of $\mu\lambda_p \geq 0$ and $\lambda_q \geq \mu\lambda_q$ follows $\Phi \geq 0$. Φ is maximal for $\lambda_p = 1$ and $\Phi(\mu, 1, \lambda_q) = \mu + \lambda_q - \mu\lambda_q \leq 1$. \square

Definition 4.19. [ZNL18] Let \mathcal{V} denote the set of vertices of a polytope P .

Remark 4.20. [ZNL18] Clearly, the Minkowski sum of two polytopes is given by the convex hull of the Minkowski sum of their vertex sets, i.e., $P_1 + P_2 = \text{Conv}(\mathcal{V}(P_1) + \mathcal{V}(P_2))$.

With this observation, the following is immediate.

Proposition 4.21. [ZNL18, p. 4] Let $f, g \in \text{Pol}(d, 1) = \mathbb{T}[x_1, \dots, x_d]$ be tropical polynomials. Then

$$\begin{aligned}\mathcal{P}(f \odot g) &= \mathcal{P}(f) + \mathcal{P}(g), \\ \mathcal{P}(f \oplus g) &= \text{Conv}(\mathcal{V}(\mathcal{P}(f)) \cup \mathcal{V}(\mathcal{P}(g))).\end{aligned}$$

We reproduce below part of [GS93, Theorem 2.1.20] and derive a corollary for bounding the number of vertices on the upper faces of a zonotope.

Theorem 4.22. [GS93]. Let P_1, \dots, P_k be polytopes in \mathbb{R}^d and let m denote the total number of non-parallel edges of P_1, \dots, P_k . Then the number of vertices of $P_1 + \dots + P_k$ does not exceed

$$\sum_{j=0}^{d-1} \binom{m-1}{j}.$$

The upper bound is attained if all P_i 's are zonotopes and all their generating line segments are in general positions.

The analysis of linear regions of neural networks, in section 5, will require an upper bound of the number of faces of a zonotope.

Corollary 4.23. [ZNL18, p. 4] Let $\mathcal{P} \in \mathbb{R}^{d+1}$ be a zonotope generated by m line segments P_1, \dots, P_m . Let $\pi : \mathbb{R}^d \times \mathbb{R} \rightarrow \mathbb{R}^d$ be the projection. Suppose \mathcal{P} satisfies:

- (i) the generating line segments are in general positions;
- (ii) the set of projected vertices $\{\pi(v) : v \in \mathcal{V}(\mathcal{P})\} \subseteq \mathbb{R}^d$ are in general position.

Then \mathcal{P} has

$$\sum_{j=0}^d \binom{m}{j}$$

vertices on its upper faces. If either (i) or (ii) is violated, then this becomes an upper bound.

Proof. Let V_1 be the vertices of the upper and V_2 the vertices of the lower face of \mathcal{P} in relation to the projection π . As a consequence of (i), P_1, \dots, P_k are non-parallel and from 4.22 it follows the number of vertices of \mathcal{P} is

$$n_1 = 2 \sum_{j=0}^d \binom{m-1}{j}.$$

Because a zonotope is convex 4.18 the upper and lower faces of P must already be all faces and $|V_1 \cup V_2| = n_1$ holds. Also because of 4.18, zonotopes are symmetric and therefore $|V_1| = |V_2|$ the upper and lower faces are of same cardinality. Let $P' := \pi(P)$ be the projection of P in \mathbb{R}^d . (ii) would not hold if $\pi(P_1), \dots, \pi(P_2)$ were not in general position and therefore P' is generated by m line segments and P' has

$$n_2 = 2 \sum_{j=0}^{d-1} \binom{m-1}{j}$$

vertices since the dimension has gone one lower. For any vertex $v \in P$, if $v \in V_1 \setminus V_2 \cup V_2 \setminus V_1$ then v can not be a vertex of P' since all of these get dropped by the projection. On the other hand $v \in (V_1 \cup V_2) \setminus (V_1 \setminus V_2 \cup V_2 \setminus V_1) = V_1 \cap V_2$ can not get dropped by the projection of P and therefore the number of vertices on P' equals $|V_1 \cap V_2|$, thus $n_2 = |V_1 \cap V_2|$ and we can calculate the number of vertices on the upper face of P :

$$\begin{aligned} |V_2| &= 0.5(|V_1 \cup V_2| - |V_1 \cap V_2|) + |V_1 \cap V_2| \\ &= 0.5(n_1 - n_2) + n_2 \\ &= 0.5(2 \sum_{j=0}^d \binom{m-1}{j} - 2 \sum_{j=0}^{d-1} \binom{m-1}{j}) + 2 \sum_{j=0}^{d-1} \binom{m-1}{j} \\ &= \binom{m-1}{d} + 2 \sum_{j=0}^{d-1} \binom{m-1}{j} \\ &= \sum_{j=0}^d (\binom{m-1}{j} + \binom{m-1}{j-1}) \\ &= \sum_{j=0}^d \binom{m}{j} \end{aligned}$$

□

5 Tropical geometry of neural networks

This is the last section of the thesis, which will provide mainly two statements about feedforward neural networks satisfying (a)–(c), proposition 5.2 and theorem 5.3. Both statements give bounds on the number of linear regions. For the first statement, proposition 5.2, we remind of the score function from remark 2.14. A neural network $\nu : \mathbb{R}^d \rightarrow \mathbb{R}^p$ together with a choice of score function $s : \mathbb{R}^p \rightarrow \mathbb{R}$ give a classifier. An input x is supposed to be classified as one of two categories, where the output value of $s(\nu(x))$ with a decision threshold c determine how x gets classified. Say $s(\nu(x)) > c$, then the neural network predicts x is from one category and if $s(\nu(x)) < c$, then from the other. As an example the input could be a picture of a MRI scanner and the categories could be cancer or no cancer.

Definition 5.1. [ZNL18] This way the input space is partitioned into two disjoint subsets by the **decision boundary** $\mathcal{B} := \{x \in \mathbb{R}^d : \nu(x) = s(c)^{-1}\}$. Connected regions with value above the threshold and connected regions with value below the threshold will be called the **positive regions** and **negative regions** respectively.

Now we will give an upper limit on the number of positive and negative regions.

Proposition 5.2. [ZNL18](Tropical geometry of decision boundary). Let $v : \mathbb{R}^d \rightarrow \mathbb{R}$ be an L -layer neural network satisfying assumptions (a)–(c) with $t^{(L)} = -\infty$. Let the score function $s : \mathbb{R} \rightarrow \mathbb{R}$ be injective with decision threshold c in its range. If $v = f \odot g$ where f and g are tropical polynomials, then

- (i) its decision boundary $\mathcal{B} = \{x \in \mathbb{R}^d : v(x) = s^{-1}(c)\}$ divides \mathbb{R}^d into at most $\mathcal{N}(f)$ connected positive regions and at most $\mathcal{N}(g)$ connected negative regions;
- (ii) if $c \in \mathbb{R}$ is such that there is no tropical monomial in $f(x)$ that differs from any tropical monomial in $g(x)$ by c , then its decision boundary is contained in the tropical hypersurface of the tropical polynomial $s^{-1}(c) \odot g(x) \oplus f(x) = \max\{f(x), g(x) + s^{-1}(c)\}$, i.e.

$$\mathcal{B} \subseteq \mathcal{T}(s^{-1}(c) \odot g \oplus f).$$

Proof. For the first statement let $s^{-1}(c) = c' \in \mathbb{R}$ and $U_1, \dots, U_{\mathcal{N}(f)}$ be the regions of \mathbb{R}^d on which f is linear. Each of the U_i is convex for $i = 1, \dots, \mathcal{N}(f)$ since if we take two elements from U_i , then all elements on the connecting line segment also have to have the same maximal monomial, or else the initial elements would have had different maximal monomials. Because of lemma 1.13 the tropical monomial $-g$ will be concave on U_i for $i = 1, \dots, \mathcal{N}(f)$ and with

$$\begin{aligned} (f \odot g)(\lambda x + (1 - \lambda)y) &= \lambda f(x) + (1 - \lambda)f(y) - g(\lambda x + (1 - \lambda)y) \\ &\geq \lambda(f \odot g)(x) + (1 - \lambda)(f \odot g)(y) \end{aligned}$$

for all $x, y \in U_i$ and $\lambda \in [0, 1] \subset \mathbb{R}$ the tropical rational function $f \odot g$ will also be concave on U_i . Now because $f \odot g$ is concave on all U_i , which themselves are convex, we can show that the set $\{x \in U_i : f \odot g(x) \geq c'\}$ must be convex:

$$\begin{aligned} (f \odot g)(\lambda x + (1 - \lambda)y) &= \lambda f(x) + (1 - \lambda)f(y) - g(\lambda x + (1 - \lambda)y) \\ &\geq \lambda(f \odot g)(x) + (1 - \lambda)(f \odot g)(y) \\ &\geq \lambda c + (1 - \lambda)c \\ &= c. \end{aligned}$$

If there were disjunct connected positive regions, then $\{x \in U_i : (f \odot g)(x) \geq c'\}$ would not be convex and therefore there can only be one connected positive region on U_i for $i = 1, \dots, \mathcal{N}(f)$.

Now if we divide \mathbb{R}^d into $\mathcal{N}(g)$ convex regions on which g is linear, then $f \odot g$ is convex on these regions and $\mathcal{N}(g)$ holds as upper bound to the connected negative regions in the same way.

For the second statement we rearrange the terms in the decision boundary $\mathcal{B} = \{x \in \mathbb{R}^d : \nu(x) = s^{-1}(c)\} = \{x \in \mathbb{R}^d : f(x) = g(x) + c'\}$ and since $f(x)$ and $g(x) + c'$ are both tropical polynomial, we have

$$\begin{aligned} f(x) &= b_1 x^{\alpha_1} \oplus \dots \oplus b_r^{\alpha_r}, \\ g(x) + c' &= c_1 x^{\beta_1} \oplus \dots \oplus c_s x^{\beta_s}, \end{aligned}$$

with appropriate multiindices $\alpha_1, \dots, \alpha_r, \beta_1, \dots, \beta_s$, and real coefficients $b_1, \dots, b_r, c_1, \dots, c_s$. By the assumption on the monomials, we have that $x_0 \in \mathcal{B}$ only if there exist i, j so that $\alpha_i \neq \beta_j$ and $b_i x_0^{\alpha_i}$. This completes the proof since if we combine the monomials of $f(x)$ and $g(x) + c'$ by (tropical) summing them into a single tropical polynomial, $\max\{f(x), g(x) + c'\}$, the above implies that on the level set, the value of the combined tropical polynomial is attained by at least two monomials and therefore $x_0 \in \mathcal{T}(\max\{f(x), g(x) + c'\}) = \mathcal{T}(s^{-1}(c) \odot g \oplus f)$. \square

The following core theorem of this section gives a general upper bound on the number of linear regions of a feedforward neural network.

Theorem 5.3. [ZNL18, p. 8] Let $\nu : \mathbb{R}^d \rightarrow \mathbb{R}$ be an L -layer real-valued feedforward neural network satisfying (a)-(c). Let $t^{(L)} = -\infty$ and $n_l \geq d$ for all $l = 1, \dots, L-1$. Then $\nu = \nu^{(L)}$ has at most

$$\prod_{l=1}^{L-1} \sum_{i=0}^d \binom{n_l}{i}$$

linear regions. In particular, if $d \leq n_1, \dots, n_{L-1} \leq n$, the number of linear regions of ν is bounded by $\mathcal{O}(n^{d(L-1)})$.

The rest of section 5 will be devoted to the proof of theorem 5.3. At first, we extend the definition of the graph of a tropical polynomial to tropical rational maps, for which linear regions do not have to be convex.

Definition 5.4. [ZNL18, p. 15] Given a tropical rational map $F \in \text{Rat}(d, m)$, we define $\mathcal{T}(F)$ to be the boundaries between adjacent linear regions. When $F = (f_1, \dots, f_m) \in \text{Pol}(d, m)$, i.e., a tropical polynomial map, this set is exactly the union of tropical hyper-surfaces $\mathcal{T}(f_i), i = 1, \dots, m$. Therefore this definition of $\mathcal{T}(F)$ extends definition 4.1.

When restricting a function $f : X \rightarrow Y$ on the domain $C \subset X$ we will write $f|_C$ for the restriction $f|_C : C \rightarrow Y$.

Definition 5.5. [ZNL18, p. 15] The **convex degree** of a tropical rational map $F \in \text{Rat}(d, n)$ is the minimum division of \mathbb{R}^d into convex regions over which F is linear, i.e.

$$\mathcal{N}_C(F) := \min\{n : C_1 \cup \dots \cup C_n = \mathbb{R}^d \text{ where } C_i \text{ convex and } F|_{C_i} \text{ linear}\}.$$

Definition 5.6. [ZNL18] For $m \leq d$ and $F \in \text{Rat}(d, n)$, we will denote by $\mathcal{N}_C(F \mid m)$ the **maximum convex degree** obtained by restricting F to an m -dimensional affine subspace in \mathbb{R}^d , i.e.,

$$\mathcal{N}_C(F \mid m) := \max\{\mathcal{N}_C(F|_\Omega) : \Omega \subseteq \mathbb{R}^d \text{ is an } m\text{-dimensional affine space}\}.$$

We will use the convention, that if $m > d$ and F is defined as in definition 5.6, then $\mathcal{N}_C(F \mid m) = \mathcal{N}_C(F)$.

Definition 5.7. [ZNL18] Consider a tropical rational map $F = (f_1, \dots, f_n) \in \text{Rat}(d, n)$ and $\alpha = (a_1, \dots, a_n) \in \mathbb{Z}^d$. Then we define the following notation:

$$F^\alpha := \alpha^T F = a_1 f_1, \dots, a_n f_n = \bigodot_{i=1}^n a_i f_i \in \text{Rat}(d, 1).$$

We want to introduce notation for the image of a function.

Definition 5.8. Let $f : X \rightarrow Y$ be a function. Then the **Image** of f is the subset $\text{Im}(f) = \{f(x) : x \in X\} \subset Y$.

Definition 5.9. [ZNL18] The exponent $\alpha = (a_1, \dots, a_n) \in \mathbb{Z}^n$ is said to be a **general exponent** of $F \in \text{Rat}(d, n)$ if the linear regions of F^α and the linear regions of F are identical.

Firstly we need to know if general exponents exist.

Lemma 5.10. [ZNL18] Let $F \in \text{Rat}(d, n)$. Then

- (i) $\mathcal{N}(F^\alpha) = \mathcal{N}(F)$ if and only if α is a general exponent;
- (ii) F has a general exponent $\alpha \in \mathbb{N}^n$.

Proof. At first, we prove (i). From definition 5.9 follows that α is in general position if the linear regions of F and F^α are identical. If the linear regions are identical, the points $x \in \mathbb{R}^d$ at which linear regions meet, and therefore where F, F^α are not differentiable, are identical. It directly follows that $\mathcal{T}(F^\alpha) = \mathcal{T}(F) \subset \mathbb{R}^d$ are identical and in conclusion the number of linear regions $\mathcal{N}(F^\alpha) = \mathcal{N}(F)$ must also be identical.

For (ii) we need analytical basics that we will not introduce extensively, but rather refer the reader to [Tao16]. We need to show that there always exists an $\alpha \in \mathbb{N}^n$ such that F^α divides its domain \mathbb{R}^d into the same set of linear regions as F . In other words, for every pair of adjacent linear regions of F , the faces in $\mathcal{T}(F)$ that separate them are also present in $\mathcal{T}(F^\alpha)$.

Let L, M be adjacent linear regions of F . Then $F|_L$ and $F|_M$ are tropical monomials where for instance $F|_L$ has the form

$$F|_L(x) = \begin{bmatrix} c_1 \odot x^{\alpha_1} \\ \vdots \\ c_n \odot x^{\alpha_n} \end{bmatrix} = \begin{bmatrix} c_1 + \alpha_{11}x_1 + \dots + \alpha_{1d}x_d \\ \vdots \\ c_n + \alpha_{n1}x_1 + \dots + \alpha_{nd}x_d \end{bmatrix}$$

and therefore the differential is of the form:

$$dF|_L = \begin{bmatrix} \alpha_{11} & \cdots & \alpha_{1d} \\ \vdots & \ddots & \vdots \\ \alpha_{n1} & \cdots & \alpha_{nd} \end{bmatrix}.$$

We see that the differentials of $F|_L, F|_M$ must have integer coefficients $dF|_L, dF|_M \in \mathbb{Z}^{nd}$ and also must be unequal, $dF|_L \neq dF|_M$, because otherwise L and M can be merged into a single region. With that in mind it is clear, that $\text{Im}(dF|_L - dF|_M) \neq \{0\}$ and the solutions set $\{\alpha \in \mathbb{N}^n : (F|_L - dF|_L)^T \alpha = 0\}$ is of dimension ≥ 1 . With the observation that $dF^\alpha|_L = \alpha^T dF|_L$:

$$dF^\alpha|_L = \begin{bmatrix} \alpha_1 a_{11} + \cdots + \alpha_n a_{n1} \\ \vdots \\ \alpha_1 a_{1d} + \cdots + \alpha_n a_{nd} \end{bmatrix} = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix}^T \cdot \begin{bmatrix} a_{11} + \cdots + a_{n1} \\ \vdots \\ a_{1d} + \cdots + a_{nd} \end{bmatrix} = \alpha^T dF|_L.$$

The same holds for $dF^\alpha|_M$. We set $\mathcal{A}(F)$ to be the collection of all pairs of adjacent linear regions of F , this set is finite. Since the set of α that degenerates two adjacent linear regions into a single one, i.e.,

$$\mathcal{S} := \bigcup_{(L,M) \in \mathcal{A}(F)} \{\alpha \in \mathbb{N}^n : (dF|_L - dF|_M)^T \alpha = 0\},$$

is contained in a union of a finite number of hyperplanes in \mathbb{R}^n , \mathcal{S} cannot cover the entire lattice of nonnegative integers \mathbb{N}^n . Therefore the set $\mathbb{N}^n \cap (\mathbb{R}^n \setminus \mathcal{S})$ is non-empty and any of its element is a general exponent for F . \square

Theorem 5.11. [ZNL18] Let $F = (f_1, \dots, f_m) \in \text{Rat}(n, m)$ and $G \in \text{Rat}(d, n)$. Define $H = (h_1, \dots, h_m) \in \text{Rat}(d, m)$ by

$$h_i := f_i \circ G, \quad i = 1, \dots, m.$$

Then

$$\mathcal{N}(H) \leq \mathcal{N}_C(H) \leq \mathcal{N}_C(F \mid d) \cdot \mathcal{N}_C(G).$$

Proof. The first inequality holds, because the regions we are counting in $\mathcal{N}_C(H)$ have one more restraint than the regions that are counted in $\mathcal{N}(H)$, that they have to be convex, and therefore $\mathcal{N}(H) \leq \mathcal{N}_C(H)$. For the second inequality we know by the definition of $\mathcal{N}_C(G)$ that there exist convex sets $C_1, \dots, C_{\mathcal{N}_C(G)}$ whose union is \mathbb{R}^d and where G is linear on each set. So $G|_{C_i}$ is some affine function ρ_i . Now we want to show for $i = 1, \dots, \mathcal{N}_C(G)$ that $\mathcal{N}_C(F \circ \rho_i) \leq \mathcal{N}_C(F \mid d)$ and first remind of the definitions:

$$\begin{aligned} \mathcal{N}_c(F \circ \rho_i) &= \min\{n : C_1 \cup \cdots \cup C_n = \mathbb{R}^d \text{ where } C_i \text{ convex and } F \circ \rho_i|_{C_i} \text{ linear}\} \\ &\leq \mathcal{N}_C(F \mid d) = \max\{\mathcal{N}_C(F|_\Omega) : \Omega \subset \mathbb{R}^n \text{ is an } d \text{ dimensional affine space}\}. \end{aligned}$$

The set $\text{Im}(\rho_i) \subset \mathbb{R}^n$ is a linear subspace, which also is an affine space. If $d < n$, then the image of ρ , $\text{Im}(\rho_i) \subset \mathbb{R}^n$ is a d -, or lower, dimensional affine space and therefore

$$\mathcal{N}_C(F \circ \rho_i) \leq \mathcal{N}_C(F \mid d).$$

On the other hand if $d \geq n$, then

$$\mathcal{N}_C(F \mid d) = \mathcal{N}_C(F \mid n) = \mathcal{N}_C(F)$$

and $\text{Im}(\rho_i)$ is a linear subspace, combined with the convexity of the sets C_i

$$\mathcal{N}_C(F \circ \rho_i) \leq \mathcal{N}_C(F \mid d)$$

holds. Since $F \circ G = F \circ \rho_i$ on C_i , we have

$$\mathcal{N}_C(F \circ G) \leq \sum_{i=1}^k \mathcal{N}_C(F \circ \rho_i).$$

Hence

$$\mathcal{N}_C(F \circ G) \leq \sum_{i=1}^k \mathcal{N}_C(F \circ \rho_i) \leq \sum_{i=1}^k \mathcal{N}_C(F \mid d) = \mathcal{N}_C(F \mid d) \cdot \mathcal{N}_C(G).$$

□

Lemma 5.12. [ZNL18] Let $\sigma^{(l)} \circ \rho^{(l)} : \mathbb{R}^{n_{l-1}} \rightarrow \mathbb{R}^{n_l}$ where $\sigma^{(l)}$ and $\rho^{(l)}$ are affine transformations and activation of the l -th layer of a neural network. If $d \leq n_l$, then

$$\mathcal{N}_C(\sigma^{(l)} \circ \rho^{(l)} \mid d) \leq \sum_{i=0}^d \binom{n_l}{i}.$$

Proof. We want to find an upper bound for $\mathcal{N}_C(\sigma^{(l)} \circ \rho^{(l)} \mid d)$ which is the maximum over the set of the convex degrees of tropical rational maps $F = (f_1, \dots, f_{n_l}) : \mathbb{R}^d \rightarrow \mathbb{R}^{n_l}$ of the form

$$f_i := \sigma_i^{(l)} \circ \rho^{(l)} \circ (b'_1 \odot x^{\alpha'_1}, \dots, b'_{n_{l-1}} \odot x^{\alpha'_{n_{l-1}}})$$

since $\{b'_1 \odot x^{\alpha'_1}, \dots, b'_{n_{l-1}} \odot x^{\alpha'_{n_{l-1}}} \mid x \in \mathbb{R}\} = \Omega$ for any $\Omega \subset \mathbb{R}^{n_{l-1}}$ and appropriate $b'_1, \dots, b'_{n_{l-1}} \in \mathbb{R}$ and $\alpha'_1, \dots, \alpha'_{n_{l-1}} \in \mathbb{R}^d$. Both functions are affine transformations so they define

$$\rho^{(l)} \circ (b'_1 \odot x^{\alpha'_1}, \dots, b'_{n_{l-1}} \odot x^{\alpha'_{n_{l-1}}}) = (b_1 \odot x^{\alpha_1}, \dots, b_{n_{l-1}} \odot x^{\alpha_{n_{l-1}}}) =: \rho'$$

a new affine linear transformation for some $b_1 \dots b_{n_{l-1}} \in \mathbb{R}$ and $\alpha_1 \dots \alpha_{n_{l-1}} \in \mathbb{R}^d$. We now have to find an upper bound on the convex degree of the neural network

$$\sigma^{(l)} \circ \rho' : \mathbb{R}^d \rightarrow \mathbb{R}^{n_l}.$$

Let $\gamma = (c_1, \dots, c_{n_l})$ be a non negative general exponent for $\sigma^{(l)} \circ \rho'$. Then

$$\sigma^{(l)} \circ \rho'(x) = \sigma^{(l)}(Ax + b) = \max\{Ax + b, t\} = \max\{A^+x + b, A^-t\} - A^-$$

and

$$\begin{aligned} \bigodot_{j=1}^{n_l} (\sigma \circ \rho')_j^{c_j}(x) &= \bigodot_{j=1}^{n_l} \left[\left(\bigodot_{i=1}^d b_i \odot x^{a_{ji}^+} \right) \oplus \left(\bigodot_{i=1}^d x^{a_{ji}^-} \right) \odot t_j \right]^{c_j} - \bigodot_{j=1}^{n_l} \left(\bigodot_{i=1}^d x^{a_{ji}^-} \right)^{c_j} \\ &=: g(x). \end{aligned}$$

The last term is linear in x , we may drop it without affecting the convex degree of the entire expression and with proposition 4.13 and 4.21 we get:

$$\begin{aligned}\mathcal{P}(g) &= \mathcal{P} \left(\bigodot_{j=1}^{n_l} \left[\left(\bigodot_{i=1}^d b_i \odot x^{a_{ji}^+} \right) \oplus \left(\bigodot_{i=1}^d x^{a_{ij}^-} \right) \odot t_j \right]^{c_j} \right) \\ &= \sum_{j=1}^{n_l} c_j \mathcal{P} \left[\left(\bigodot_{i=1}^d b_i \odot x^{a_{ji}^+} \right) \oplus \left(\bigodot_{i=1}^d x^{a_{ij}^-} \right) \odot t_j \right] \\ &= \sum_{j=1}^{n_l} c_j \text{Conv}(\mathcal{V}(\mathcal{P}(f_j)) \cup \mathcal{V}(\mathcal{P}(g_j)))\end{aligned}$$

where

$$f_j(x) = \bigodot_{i=1}^d b_i \odot x^{a_{ji}^+} \quad \text{and} \quad g_j(x) = \left(\bigodot_{i=1}^d x^{a_{ij}^-} \right) \odot t_j.$$

For the monomials f_j, g_j , the sets $\mathcal{P}(f_j)$ and $\mathcal{P}(g_j)$ are points in \mathbb{R}^{d+1} and $\text{Conv}(\mathcal{V}(\mathcal{P}(f_j)) \cup \mathcal{V}(\mathcal{P}(g_j)))$ is a line in \mathbb{R}^{d+1} . Hence g is a Minkowski sum of n_l line segments in \mathbb{R}^{d+1} . The generating line segments are not necessarily in general position thus corollary 4.23 only gives an upper bound:

$$\mathcal{P}(g) \leq \sum_{i=0}^d \binom{n_l}{i}.$$

This means, that

$$\mathcal{N}_C(\sigma^{(l)} \circ \rho^{(l)} \Big|_{\Omega}) \leq \sum_{i=0}^d \binom{n_l}{i}$$

for all $\Omega \subset \mathbb{R}^d$ and so

$$\begin{aligned}\mathcal{N}_C(\sigma^{(l)} \circ \rho^{(l)} \mid d) &= \max\{\mathcal{N}_C(\sigma^{(l)} \circ \rho^{(l)} \Big|_{\Omega}) : \Omega \subset \mathbb{R}^{n_l-1} \text{ is an } m\text{-dimensional affine space}\} \\ &\leq \sum_{i=0}^d \binom{n_l}{i}.\end{aligned}$$

□

Theorem 5.13. [ZNL18] Let $\nu : \mathbb{R}^d \rightarrow \mathbb{R}^{n_L}$ be an L -layer neural network satisfying assumptions (a)-(c) with $F^{(l)}, G^{(l)}, H^{(l)}$, and $\nu^{(l)}$ as defined in the proof of theorem 3.3. Let $n_l \geq d$ for all $l = 1, \dots, L$. Then

$$\mathcal{N}_C(\nu^{(1)}) = \mathcal{N}_C(G^{(1)}) = \mathcal{N}_C(H^{(1)}) = 1$$

and

$$\mathcal{N}_C(\nu^{(l+1)}) \leq \mathcal{N}_C(\nu^{(l)}) \cdot \sum_{i=0}^d \binom{n_{l+1}}{i}.$$

Proof. For $l = 1$ the functions $G^{(1)}(x) = A_-^{(1)}x$ and $H^{(1)}(x) = A_+^{(1)}x + b^{(1)}$ are linear, which forces $\mathcal{N}_C(\nu^{(1)}) = 1$. Since $\nu^{(l)} = (\sigma^{(l)} \circ \rho^{(l)}) \circ \nu^{(l-1)}$, the recursive bound follows directly from theorem 5.11 and lemma 5.12. □

Theorem 5.3 follows by induction from theorem 5.13. Also if $d \leq n_1, \dots, n_{L-1} \leq n$, then

$$\prod_{l=1}^{L-1} \sum_{i=0}^d \binom{n}{i} \leq \prod_{l=1}^{L-1} n^d = n^{d(L-1)}.$$

With the lower bound $\Omega((n/d)^{(L-1)d}n^d)$, from [MPCB14, corollary 5] cited in [MS15], we can suspect that the number of linear regions of a feedforward neural network grows polynomially with the width n and exponentially with the number of layers L .

6 Conclusion

In section 1 and 2, we introduced basic concepts of tropical algebra and neural networks by extending the knowledge of [ZNL18] and adding examples. Then we successfully linked feedforward neural networks under assumptions (a)-(c) to tropical rational maps in section 3, theorem 3.3, then further restricted both to a single output, which led to the equivalence between tropical rational functions, continuous piecewise linear functions with integer coefficients and neural networks satisfying assumptions (a)-(c). In the last two sections, section 4 and 5, we then proved upper bounds on the linear regions of neural networks satisfying assumptions (a)-(c).

Thus we have successfully proven an equivalence between feedforward neural networks and tropical rational functions and used the knowledge to find upper limits to the number of regions of a neural network. On the basis of the estimates we have then phrased a conjecture where the number of linear regions of a feedforward neural network grows polynomially with the width and exponentially with the number of Layers. This is not subject to this thesis and may be further researched.

As the thesis has followed the paper [ZNL18] that introduces the fact that feedforward neural networks are nothing more than tropical rational maps, the acquired knowledge may be used to apply notions of tropical geometry to give alternative proves to already known facts on neural networks and possibly also new insights.

References

- [Abe] Seth Abels, *Boolean gates truth tables*, https://www.researchgate.net/figure/Summary-of-the-common-Boolean-logic-gates-with-symbols-and-truth-tables_fig3_291418819, Accessed: 2020-08-24.
- [AHSB14] Forest Agostinelli, Matthew Hoffman, Peter Sadowski, and Pierre Baldi, *Learning activation functions to improve deep neural networks*, arXiv preprint arXiv:1412.6830 (2014).
- [Ale] AlexMartinezMingo, *weighted graph exercise*, <https://imgur.com/gallery/eTBto79>, Accessed: 2020-08-19.
- [BBV04] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe, *Convex optimization*, Cambridge university press, 2004.
- [BG14] Annemarie Borg and Reinhold Gschweicher, *Script 2013w 104.271 discrete mathematics vo (gittenberger)*.
- [Bis06] Christopher M Bishop, *Pattern recognition and machine learning*, springer, 2006.
- [BP85] Jean Berstel and Dominique Perrin, *Theory of codes*, Academic Press, 1985.
- [Bry61] Arthur E Bryson, *A gradient method for optimizing multi-stage allocation processes*, Proc. Harvard Univ. Symposium on digital computers and their applications, vol. 72, 1961.
- [BW10] Edward A Bender and S Gill Williamson, *Lists, decisions and graphs*, S. Gill Williamson, 2010.
- [CMGS10] Dan Claudiu Cireşan, Ueli Meier, Luca Maria Gambardella, and Jürgen Schmidhuber, *Deep, big, simple neural nets for handwritten digit recognition*, Neural computation **22** (2010), no. 12, 3207–3220.
- [GC] Balaji Venkateswaran Giuseppe Ciaburro, *Neural networks with r*, <https://static.packt-cdn.com/products/9781788397872/graphics/1ebc2a0a-2123-4351-b7e1-eb57f098bafa.png>, Accessed: 2020-08-19.
- [GS93] Peter Gritzmann and Bernd Sturmfels, *Minkowski addition of polytopes: computational complexity and applications to gröbner bases*, SIAM Journal on Discrete Mathematics **6** (1993), no. 2, 246–269.
- [Jar] Quasar Jarosz, *Depiction of a neuron*, https://commons.wikimedia.org/wiki/File:Neuron_Hand-tuned.svg, Accessed: 2020-08-18.
- [Kel60] Henry J Kelley, *Gradient theory of optimal flight paths*, Ars Journal **30** (1960), no. 10, 947–954.
- [Kle09] Paul Klemperer, *A new auction for substitutes: Central-bank liquidity auctions, toxic asset auctions, and variable product-mix auctions*, Journal of the European Economic Association, Forthcoming (2009).

- [Kri14] Nikolai Krivulin, *Tropical optimization problems*, arXiv preprint arXiv:1408.0313 (2014).
- [LMD⁺11] Quoc V. Le, Rajat Monga, Matthieu Devin, Greg Corrado, Kai Chen, Marc’Aurelio Ranzato, Jeffrey Dean, and Andrew Y. Ng, *Building high-level features using large scale unsupervised learning*, CoRR **abs/1112.6209** (2011).
- [Mas85] Victor P Maslov, *On a new superposition principle for optimization problem*, Séminaire Équations aux dérivées partielles (Polytechnique) (1985), 1–14.
- [Mik05] Grigory Mikhalkin, *Enumerative tropical algebraic geometry in \mathbb{R}^2* , Journal of the American Mathematical Society **18** (2005), no. 2, 313–377.
- [MP43] Warren S McCulloch and Walter Pitts, *A logical calculus of the ideas immanent in nervous activity*, The bulletin of mathematical biophysics **5** (1943), no. 4, 115–133.
- [MPCB14] Guido F Montufar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio, *On the number of linear regions of deep neural networks*, Advances in neural information processing systems, 2014, pp. 2924–2932.
- [MS15] Diane Maclagan and Bernd Sturmfels, *Introduction to tropical geometry*, vol. 161, American Mathematical Soc., 2015.
- [Nie17] Michael Nielsen, *How the backpropagation algorithm works*, Dosegljivo: <http://neuralnetworksanddeeplearning.com/chap2.html> (2017).
- [OGAJ20] Fateme Olia, Shaban Ghalandarzadeh, Amirhossein Amiraslani, and Sedighe Jamshidvand, *Analysis of linear systems over idempotent semifields*, Mathematical Sciences **14** (2020), no. 2, 137–146.
- [Pal56] Sanford L Palay, *Synapses in the central nervous system*, The Journal of biophysical and biochemical cytology **2** (1956), no. 4, 193.
- [Qui55] Willard V Quine, *A way to simplify truth functions*, The American mathematical monthly **62** (1955), no. 9, 627–631.
- [RHW86] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams, *Learning representations by back-propagating errors*, nature **323** (1986), no. 6088, 533–536.
- [Tao16] Terence Tao, *Analysis ii*, Springer Singapore, 2016.
- [TM99] JM Tarela and MV Martinez, *Region configurations for realizability of lattice piecewise-linear models*, Mathematical and Computer Modelling **30** (1999), no. 11-12, 17–27.
- [WH60] Bernard Widrow and Marcian E Hoff, *Adaptive switching circuits*, Tech. report, Stanford Univ Ca Stanford Electronics Labs, 1960.
- [ZNL18] Liwen Zhang, Gregory Naitzat, and Lek-Heng Lim, *Tropical geometry of deep neural networks*, arXiv preprint arXiv:1805.07091 (2018).

Eidesstattliche Erklärung

Ich versichere an Eides statt durch meine eigene Unterschrift, dass ich die vorstehende Arbeit selbständig und ohne fremde Hilfe angefertigt und alle Stellen, die aus Veröffentlichungen genommen sind, als solche kenntlich gemacht habe. Die Versicherung bezieht sich auch auf in der Arbeit gelieferte Zeichnungen, Skizzen, bildliche Darstellungen und dergleichen.

Ort, Datum

Unterschrift