# A NOTATION

| Symbol | Meaning |
|---|---|
| $d, d_i$ | vector dimension of layers |
| $s$ | number of labels for classification |
| $\ell$ | linear layer of a neural network, equal to a function $\mathbb{R}^{d_1} \to \mathbb{R}^{d_2}$ |
| $\nu$ | tropical rational function $\mathbb{R}^d \to \mathbb{R}$ |
| $(\nu_i)_i$ | tropical rational map $\mathbb{R}^d \to \mathbb{R}^s$ |
| $\mathbf{W}$ | weight matrix |
| $\mathbf{W}_j$ | $j^{\text{th}}$ row of $\mathbf{W}$ |
| $\mathbf{W}_{\bullet i}$ | $i^{\text{th}}$ column of a weight matrix $\mathbf{W}$ |
| $\mathbf{W}^{\text{pos}}$ | positive part of a matrix defined by entries $w_{ij}^{\text{pos}} = \max\{0, w_{ij}\}$ |
| $\mathbf{W}^{\text{neg}}$ | negative part of a matrix defined by entries $w_{ij}^{\text{neg}} = \max\{0, -w_{ij}\}$ |
| $\mathbf{c}$ | bias vector |
| $\sigma$ | activation function |
| $\mathcal{N}$ | neural network function $\mathcal{N}$ |
| $\mathbf{x}$ | vector describing data sample or input to layer of a neural network |
| $x_j$ | vector components of $\mathbf{x}$ in $\mathbb{R}$ |
| $\mathcal{X}, N$ | $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ data set of $N$ samples |
| $\mathbf{y}$ | output of layer or activation of neural network, $\mathbf{y} = \mathbf{W}\mathbf{x} + \mathbf{c}$ or $\mathbf{y} = \sigma(\mathbf{x})$ |
| $p, q$ | tropical polynomial |
| $\frac{p}{q}$ | tropical rational function |
| $p_{\text{eval}}, \left(\frac{p}{q}\right)_{\text{eval}}$ | evaluation of tropical polynomials/rational functions |
| $\mathbb{R}$ | real numbers |
| $\mathbb{R}_{\geq 0}$ | non-negative real numbers |
| $\mathbb{T}$ | tropical semiring, $(\mathbb{R} \cup \{-\infty\}, \oplus, \odot)$ |
| $\mathbb{T}[x_1, \ldots, x_n]$ | semiring of tropical polynomials in $n$ variables |
| $\mathbb{T}(x_1, \ldots, x_n)$ | semifield of tropical rational functions in $n$ variables |
| $(\mathbf{A}, \mathbf{a}), (\mathbf{B}, \mathbf{b})$ | matrix representation of a tropical polynomial |
| $\oplus$ | tropical addition, tropical matrix addition |
| $\odot$ | tropical multiplication, tropical matrix multiplication |
| $\oslash$ or $\frac{\bullet}{\bullet}$ | tropical division, tropical matrix division |
| $(\mathbf{A}, \mathbf{a})^s$ | tropical array-vector pair $(\mathbf{A}, \mathbf{a})$ to the tropical power of $s$ |
| $(\mathbf{A}^+, \mathbf{a}^+)/(\mathbf{A}^-, \mathbf{a}^-)$ | matrix representation of a tropical rational function |
| $(\nu_{\mathbf{j}})_{\mathbf{j}}$ | list of tropical array-vector-pairs indexed by subscripts |
| $\left(\frac{\nu_i^+}{\nu_i^-}\right)_{\mathbf{j}}$ | array of tropical rational functions representing a tropical rational map |
| $\mathbf{A}_{i\bullet}$ | $i^{\text{th}}$ row of $\mathbf{A}$ |
| $\mathbf{A}_{\bullet 1:d}$ | first $d$ columns of $\mathbf{A}$ |
| $\nu_{\text{eval}}, \left(\frac{\nu^+}{\nu^-}\right)_{\text{eval}}$ | tropical evaluation of tropical matrices |
| $\nu(\mathbf{x})$ | evaluation of $\nu$ at $\mathbf{x}$, equal to $\nu_{\text{eval}}(\mathbf{x})$ |
| $\mu \circ \nu$ | tropical function representing the composition $\mu_{\text{eval}} \circ \nu_{\text{eval}}$ |
| $[\mathbf{A}]_{\mathbf{x}}, [\mathbf{a}]_{\mathbf{x}}$ | rows of $\mathbf{A}$ and $\mathbf{a}$, respectively, whose evaluations on $\mathbf{x}$ are maximal |
| $\mathbf{A}_{\max}$ | row vector of the maxima of all but the last column of a pair of tropical matrices $\mathbf{A}^+, \mathbf{A}^-$ |
| $\mathbf{K}$ | row vector in $\mathbb{R}^{d_1}$ ensuring that a fraction of trop matrices is represented in $\mathbf{T}_{d_1}^+$ |
| $\mathbf{T}_d$ | set of array-vector pairs, where the array contains only positive elements. The product of the number of dimensions (apart from the first one) of of the array is equal to $d$. |
| $\sim$ | equivalence relation on $\mathbf{T}_d$ defined by equivalence of the evaluation function |
| $\mathbb{T}_d$ | $\mathbf{T}/\sim$ |
| $\text{Frac}(\mathbb{T}_d)$ | (semi-) field of fractions of $\mathbb{T}_d$ |

# B EQUIVALENCE BETWEEN TROPICAL MATRICES AND TROPICAL POLYNOMIALS

## B.1 BASICS OF TROPICAL ALGEBRA

We introduce necessary notions of tropical algebra, an area previously connected with ReLU networks by (Zhang et al., 2018; Charisopoulos & Maragos, 2018). A more detailed basic introduction to tropical algebra can be found there.

**Definition B.1.** The **tropical semiring** $\mathbb{T}$ is the triple $\mathbb{T} := \{\mathbb{R} \cup \{-\infty\}, \oplus, \odot\}$, where $\oplus$ and $\odot$ denote tropical addition (ordinary maximum) and tropical multiplication (ordinary addition), respectively.
*In ordinary notation,* $1 \odot 2^3 \oplus 4 \odot 5^6$ *just means* $\max\{1 + 3 \cdot 2, 4 + 6 \cdot 5\} = \max\{7, 34\} = 34$.

**Definition B.2.** A (generalized)[7] **tropical monomial** in $d$ variables $x_1, \ldots, x_d$ is an expression of the form $c \odot x_1^{a_1} \odot x_2^{a_2} \odot \cdots \odot x_d^{a_d}$ where $c, a_1, \ldots, a_d \in \mathbb{R}_{\geq 0}$. As a convenient shorthand, we will also write a tropical monomial as $cx_1^{a_1}x_2^{a_2}\ldots x_d^{a_d}$ and in multiindex notation as $c\mathbf{x}^\alpha$, where $\alpha = (a_1, \ldots, a_d) \in \mathbb{R}^d_{\geq 0}$ and $\mathbf{x} = (x_1, \ldots, x_d)$. Note that $\mathbf{x}^\alpha = 0 \odot \mathbf{x}^\alpha$ as 0 is the **tropical multiplicative identity**.
*In ordinary notation,* $c \odot x_1^{a_1} \odot x_2^{a_2} \odot \cdots \odot x_d^{a_d}$ *is a linear function* $c + a_1x_1 + a_2x_2 + \cdots + a_dx_d$.

**Definition B.3.** A (generalized)[4] **tropical polynomial** $p(\mathbf{x}) = p(x_1, \ldots, x_d)$ is a finite tropical sum of tropical monomials

$$p(x) = c_1\mathbf{x}^{\alpha_1} \oplus \ldots \oplus c_r\mathbf{x}^{\alpha_r}, \tag{4}$$

where $\alpha_i = (a_{i1}, \ldots, a_{id}) \in \mathbb{R}^d_{\geq 0}$ and $c_i \in \mathbb{T}$ for $1 \leq i \leq r$. 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 $i \neq j$.
*In ordinary notation, a tropical polynomial is a maximum of linear functions.*

**Definition B.4.** A **tropical rational function** is a standard difference, or, equivalently, a tropical quotient of two tropical polynomials $p(x)$ and $q(x)$:

$$p(x) - q(x) = p(x) \oslash q(x).$$

We will denote a tropical rational function by $\nu = p \oslash q$, where $p$ and $q$ are understood to be tropical polynomial functions.
*In ordinary notation, a tropical rational function is a difference of two maxima of linear functions.*

**Definition B.5.** If $\nu : \mathbb{R}^d \to \mathbb{R}^s$ is given by $x = (x_1, \ldots, x_d) \to (\nu_1(x), \ldots, \nu_s(x))$, where each $\nu_i$ is a tropical rational function, then we call $\nu$ a *tropical rational map*.

**Notation B.6.** Let $p$ denote a tropical polynomial. We denote its evaluation function $\mathbb{R}^d \to \mathbb{R}$ by $p_{eval}$. We say that two tropical polynomials $p, p'$ are equal if and only if their evaluation functions are equal. Two tropical rational functions $p \oslash q$ and $p' \oslash q'$ are defined to be equal if and only if $p \odot q' = q \odot p'$.

## B.2 TROPICAL MATRIX OPERATIONS

In this section, we formally define **tropical matrices** equipped with tropical operations as an efficient representation of tropical polynomials. This allows us to derive Table 1 and Algorithm 3.1. Therefore, **this section lays the conceptual framework and notations for all proofs in these supplements.** A lookup table for our choices of notation can be found in Section A.

Indeed, it turns out that we can treat tropical polynomials like matrix-vector pairs if we define *tropical matrix addition* and *tropical matrix multiplication* in a certain way. A map from $\mathbb{R}^d \to \mathbb{R}^k$ will be represented by a list of $k$ many pairs of tropical matrices of type $r \times d$ together with $k$ many pairs of vectors in $\mathbb{R}^r$ encoding possible bias terms. Since some neural network layers such as convolutional layers or maxpooling activation operate on multi-dimensional arrays instead of vectors, we will more generally define the notions for multi-dimensional arrays in $\mathbb{R}^{r \times n_1 \times \cdots \times n_t}_{\geq 0}$ and refer to the first dimension as "**rows**". The possibility to equivalently consider flattened versions of input and output to layers justifies the terminology of tropical "matrices" also in this general case. We will at first present the definitions and then show that the representation with tropical arrays is equivalent to the polynomial representation.

[7] We note that our terminology diverges from the one used in Zhang et al. (2018), where only integer exponents are allowed.

**Notation B.7.** Let us fix some $d \in \mathbb{N}$. We denote by $\mathbf{T}_d$ the set of pairs $\nu = (\mathbf{A}, \mathbf{a})$ with $\mathbf{A} \in \mathbb{R}_{\geq 0}^{r \times n_1 \times \cdots \times n_t}$, $\mathbf{a} \in \mathbb{R}^r$ for some $r$ and with $d = n_1 \cdot \cdots \cdot n_t$.

**Definition B.8.** Let $\nu = (\mathbf{A}, \mathbf{a}), \mu = (\mathbf{B}, \mathbf{b}) \in \mathbf{T}_d$ have with $r_1$ and $r_2$ rows, respectively. We define *tropical matrix addition* $\oplus : \mathbf{T}_d \times \mathbf{T}_d \to \mathbf{T}_d$ by $\nu \oplus \mu := (\mathbf{A} \oplus \mathbf{B}, \mathbf{a} \oplus \mathbf{b})$, where

$$\mathbf{A} \oplus \mathbf{B} := \begin{pmatrix} \mathbf{A} \\ \mathbf{B} \end{pmatrix}; \quad \mathbf{a} \oplus \mathbf{b} := \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix}.$$

For any array $\mathbf{A} \in \mathbb{R}_{\geq 0}^{r \times n_1 \times \cdots \times n_t}$, denote by $\mathbf{A}_{i\bullet} \in \mathbb{R}_{\geq 0}^{n_1 \times \cdots \times n_t}$ the sub-array corresponding to index $i$ in the first dimension. We define *tropical matrix multiplication* $\odot : \mathbf{T}_d \times \mathbf{T}_d \to \mathbf{T}_d$ by

$$(\mathbf{A} \odot \mathbf{B})_{i+(j-1)r_1\bullet} := \mathbf{A}_{i\bullet} + \mathbf{B}_{j\bullet}, \quad (\mathbf{a} \odot \mathbf{b})_{i+(j-1)r_1\bullet} := \mathbf{a}_{i\bullet} + \mathbf{b}_{j\bullet}; \quad 1 \leq i \leq r_1, 1 \leq j \leq r_2.$$

For any $s \in \mathbb{R}_{\geq 0}$, we define the *tropical power of a matrix* by

$$(\mathbf{A}, \mathbf{a})^s := (s \cdot \mathbf{A}, s \cdot \mathbf{a})$$

where $\cdot$ denotes the usual scalar multiplication.

The definition of tropical matrix addition says that $\mathbf{A} \oplus \mathbf{B} \in \mathbb{R}_{\geq 0}^{(r_1+r_2) \times n_1 \times \cdots \times n_t}$ is obtained by stacking $\mathbf{A}$ and $\mathbf{B}$ along the first dimension. The first dimension of $\mathbf{A} \oplus \mathbf{B}$ is $r_1 + r_2$. The definition of tropical matrix multiplication says that $\mathbf{A} \odot \mathbf{B} \in \mathbb{R}_{\geq 0}^{(r_1 r_2) \times n_1 \times \cdots \times n_t}$ is the array for which the new rows are given by the sums of any rows of $\mathbf{A}$ with any row of $\mathbf{B}$. The first dimension of $\mathbf{A} \odot \mathbf{B}$ is $r_1 \cdot r_2$.

**Definition B.9.** A semiring (resp. semifield) has the same algebraic structure as a ring (resp. field) but without assuming the existence of additive inverses.

**Lemma B.10.** *The sets $\mathbf{T}_d$ together with tropical matrix addition $\oplus$ and tropical matrix multiplication $\odot$ forms a semiring.*

### B.3 Tropical Evaluation

We will now describe how to evaluate matrix-vector pairs in $\mathbf{T}_d$.

**Definition B.11.** Let $\nu = (\mathbf{A}, \mathbf{a}) \in \mathbf{T}_d$. We define the *tropical evaluation* $\nu_{\text{eval}} : \mathbb{R}_{\geq 0}^{n_1 \times \cdots \times n_t} \to \mathbb{R}$ of $\nu$ at a point $\mathbf{x} \in \mathbb{R}^{n_1 \times \cdots \times n_t}$ by $\mathbf{x} \mapsto \max\{\mathbf{A}_{\text{flat}} \mathbf{x}_{\text{flat}} + \mathbf{a}\}$, where $\mathbf{A}_{\text{flat}} \in \mathbb{R}_{\geq 0}^{r \times (n_1 \cdots n_t)}$, $\mathbf{x}_{\text{flat}} \in \mathbb{R}^{n_1 \cdots n_t}$ are reshaped versions of the array and the data point and where the maximum is going over the $r$-many rows of the column vector $\mathbf{A}_{\text{flat}} \mathbf{x}_{\text{flat}} + \mathbf{a}$. For a tropical quotient $\nu = \nu^+ / \nu^-$, we define $\left(\frac{\nu^+}{\nu^-}\right)_{eval} : \mathbb{R}^d \to \mathbb{R}$ by $\mathbf{x} \mapsto \nu^+_{eval}(\mathbf{x}) - \nu^-_{eval}(\mathbf{x})$.

In the following, we will simply write $\nu(\mathbf{x})$ for $\nu_{eval}(\mathbf{x})$.

### B.4 The Semiring of Tropical Matrices

We will now define an equivalence relation that helps to represent tropical polynomials as matrices:

**Definition B.12.** We define an equivalence relation $\sim$ on $\mathbf{T}_d$ by $\nu \sim \mu$ if and only if $\nu_{eval} = \mu_{eval}$. We let $\mathbb{T}_d := \mathbf{T}_d / \sim$ be the set of *tropical matrices*. We will call an equivalence class of matrix-vector pairs a **tropical matrix**.

The equivalence relation identifies matrices that are equal up to reordering of rows and removal of rows that occur more than once, but less obvious relations exist. For example, we have that

$$\left(\begin{pmatrix} 2 & 2 \\ 4 & 0 \end{pmatrix}, \begin{pmatrix} 2 \\ 0 \end{pmatrix}\right) \sim \left(\begin{pmatrix} 2 & 2 \\ 4 & 0 \\ 3 & 1 \end{pmatrix}, \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix}\right)$$

since the last rows on the right hand side will never be maximal during evaluation ($3x + y + 1 > 2x + 2y + 2$ and $3x + y + 1 > 4x$ if and only if $x - y - 1 > 0$ and $0 > x - y - 1$).

**Lemma B.13.** *The sets $\mathbb{T}_d$ inherits well-defined operations $\oplus$ and $\odot$ from $\mathbf{T}_d$ respectively and forms a semiring under these operations.*

*Proof.* It is straightforward to show that the map $\mathbf{T}_d \to \mathbb{T}_d,\ \nu \mapsto [\nu]$ sending each pair to its equivalence class is a semiring-homomorphism. □

We will, by a slight abuse of notation, also call the operations on $\mathbb{T}_d$ tropical matrix addition and tropical matrix multiplication. We also note that if $s$ is a positive integer, then tropical power on $\mathbb{T}_d$ agrees with the definition from tropical matrix multiplication, i.e. $\nu^2 = \nu \odot \nu$ (and similarly for higher powers).

We denote the semiring of tropical polynomials in $d$ variables as in Definition B.3 by $\mathbb{T}[x_1, \ldots, x_d]$ and the semifield of tropical rational functions by $\mathbb{T}(x_1, \ldots, x_d)$. By definition, two tropical polynomials are equal if and only if they are equal as functions. The following theorem motivates the definition of $\mathbb{T}_d$.

**Theorem B.14.** *There is an isomorphism $f : \mathbb{T}_d \xrightarrow{\sim} \mathbb{T}[x_1, \ldots, x_d]$ of semirings.*

*Proof.* Let $(\mathbf{A} = [A_{ij}], \mathbf{a} = [a_i]) \in \mathbf{T}_d$. Let us define $f$ by $[(\mathbf{A}, \mathbf{a})] \mapsto \bigoplus_i a_i x_1^{A_{i1}} \cdots x_d^{A_{id}}$. The check that this map is a well-defined isomorphism from $\mathbb{T}_d^+$ to $\mathbb{T}[x_1, \ldots, x_d]$ is routine and therefore left to the reader.

The inverse map is given as follows: Let $f(x) = a_1 x^{\alpha_1} \oplus \ldots \oplus a_r x^{\alpha_r}$ denote a tropical polynomial as in (4) where $\alpha_i = (A_{i1}, \ldots, A_{id}) \in \mathbb{R}^d_{\geq 0}$ and $a_i \in \mathbb{R}$. Then we represent $f$ as a pair $(\mathbf{A}, \mathbf{a}) \in \mathbf{T}_d$. □

The theorem above allows us to treat tropical polynomials as matrix-vector pairs equipped with simple operations, which gives us an efficient representation on the computer.

**Corollary B.15.** *There is an isomorphism $\mathbb{T}(x_1, \ldots, x_d) \to \mathrm{Frac}(\mathbb{T}_d)$ between the tropical rational functions and the semifield of fractions of $\mathbb{T}_d$.*

The following lemma is easily verified from the definitions.

**Lemma B.16.** *Let $p$ be a tropical polynomial and let $\nu \in \mathbb{T}_d$ be its corresponding tropical matrix. Then $p_{eval} = \nu_{eval}$. An equivalent statement holds for rational functions and fractions of tropical matrices.*

**Notation B.17.** We will denote fractions of tropical matrices in $\mathrm{Frac}(\mathbb{T}_d)$ by $\nu^+ \oslash \nu^-$ or simply $\frac{\nu^+}{\nu^-}$ for $\nu^+, \nu^- \in \mathbb{T}_d$. Given a list of fractions of tropical matrices $\left(\frac{\nu_1^+}{\nu_1^-}, \ldots, \frac{\nu_m^+}{\nu_m^-}\right)$, we will write $\left(\frac{\nu_i^+}{\nu_i^-}\right)_i(\mathbf{x})$ to denote $\left(\frac{\nu_1^+}{\nu_1^-}(\mathbf{x}), \ldots, \frac{\nu_m^+}{\nu_m^-}(\mathbf{x})\right)$, which defines a tropical rational map (Def B.5) with the help of Lemma B.16. More generally, if $f$ is a map into $\mathbb{R}^{n_1 \times \cdots \times n_t}$, then we similarly write $\frac{\nu_{\mathbf{j}}^+}{\nu_{\mathbf{j}}^-}$ with multi-index $\mathbf{j} = (j_1, \ldots, j_t)$.

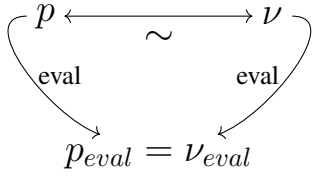



Figure 6: The relation between polynomials, matrices and their evaluations. The isomorphism between polynomials and matrices respects the evaluations.

| **Function** $f$ | **Polynomial** $p$ | **Matrix** $\nu = (\mathbf{A}, \mathbf{a})$ |
|---|---|---|
| $\mathbb{T}^d \to \mathbb{T}$ | $d$ variables | $\mathbf{A}$ has $d$ columns |
| $\mathbb{T}^d \to \mathbb{T}$ | $p \oslash q$ | $\nu^+ \oslash \nu^-$ |
| $\mathbb{T}^d \to \mathbb{T}^m$ | $(\frac{p_1}{q_1}, \ldots, \frac{p_m}{q_m})$ | $(\frac{\nu_1^+}{\nu_1^-}, \ldots, \frac{\nu_m^+}{\nu_m^-})$ |
| $\max\{f_1, f_2\}$ | $p_1 \oplus p_2$ | $\nu_1 \oplus \nu_2$ |
| $f_1 + f_2$ | $p_1 \odot p_2$ | $\nu_1 \odot \nu_2$ |
| $s \cdot f$ | $p^s$ | $\nu^s$ |
| Add. Identity | $-\infty$ | $(-\infty \quad \ldots \quad -\infty)$ |
| Mult. Identity | $0$ | $(0 \quad \ldots \quad 0)$ |

Figure 7: The correspondences between polynomials and matrices and the functions on the tropical semiring $\mathbb{T} = \{-\infty\} \cup \mathbb{R}$ they represent.

# C Derivation of the Algorithm

## C.1 Introduction

Let $\mathcal{N} = (\mathcal{N}_i)_{1\leq i\leq s}$ denote a neural network for classification into $s$ classes. Then $\max_i \mathcal{N}_i(\mathbf{x})$ is a tropical rational function by the results of (Zhang et al., 2018). Thus it can be written as

$$\mathcal{N}_i(\mathbf{x}) = \max\{a_{i1}^+(\mathbf{x}), \dots, a_{in}^+(\mathbf{x})\} - \max\{a_{i1}^-(\mathbf{x}), \dots, a_{im}^-(\mathbf{x})\},$$

with affine functions $a_{ij}^+, a_{ij}^-$ or, using Section B, as a fraction of positive tropical matrices $(\frac{\nu_i^+}{\nu_i^-})_{1\leq i\leq s}$. This section contains the derivation of the algorithm for (i) finding this representation and (ii) extracting the linear terms (or rows of the tropical matrices) that are maximized (or used in the evaluation) on $N$ training points in a data set $\mathcal{X}$.

The general strategy for the derivation of the algorithm is as follows: Using Section B, we encode each activation in a network as a tropical rational map, i.e., in terms of fractions of tropical matrices in $\mathbf{T}_d$. Then we take the last layer and represent the maximal entry of the network output as the evaluation function of a fraction of tropical matrices. We additionally keep track of the index of the maximal entry. Then, we iteratively merge earlier layers to a common representation as a tropical rational function. This leads to the whole network being represented by a fraction of tropical matrices. This theoretically possible representation, however, is not practically feasible due to the gigantic number of linear regions of a network. Therefore, we show that it is possible to extract only those rows used in the tropical evaluation on specific points in a given data set. In other words, following Definition B.11 of tropical evaluation, we extract those rows of the positive tropical matrices that are maximal on the points in the data set. The key insight allowing this extraction is that it can happen layer-wise during the merging operation. This section is organized as follows:

- Section C.2 proves the preliminary Lemma 3.2 presented in the main part of the paper.
- Section C.3 describes how to describe (Leaky) ReLU and Maxpooling activation as a tropical rational map.
- Section C.4 shows how to merge a layer with a tropical rational function that represents all later layers. The derived computations are such that a successive use of merging operations is possible.
- Section C.5 shows that a selection of rows is possible while merging layers. The necessary calculations to find suitable rows are also derived.
- Section C.6 treats the last layer.
- Finally, Section C.7 puts things together, proving Theorem 3.3 that Algorithm 3.1 performs the desired extraction of linear terms from a representation of the network function as a topical rational function.

## C.2 Proof of Lemma 3.2

As a first application of the framework of tropical matrices with tropical operations, we prove Lemma 3.2.

**Lemma 3.2** Let $\mathcal{N} : \mathbb{R}^d \to \mathbb{R}^s$ be the function of a ReLU neural network for classification with $s$ output neurons. Then there are affine functions $a_{ij}^+, a_j^-$ such that

$$\mathcal{N}(\mathbf{x}) = \begin{pmatrix} \max\{a_{11}^+(\mathbf{x}), \dots, a_{1n_1}^+(\mathbf{x})\} \\ \vdots \\ \max\{a_{1s}^+(\mathbf{x}), \dots, a_{1n_s}^+(\mathbf{x})\} \end{pmatrix} - \max\{a_1^-(\mathbf{x}), \dots, a_m^-(\mathbf{x})\},$$

where the maxima of the $a_i^-(\mathbf{x})$ on the right is subtracted from each entry of the vector on the left.

*Proof.* From (Zhang et al., 2018) we have that each component $\mathcal{N}_i$ of $\mathcal{N}$ has a representation $\mathcal{N}_i(\mathbf{x}) = \max\{a_{i1}^+(\mathbf{x}), \dots, a_{in}^+(\mathbf{x})\} - \max\{a_{i1}^-(\mathbf{x}), \dots, a_{im}^-(\mathbf{x})\}$, with linear functions $a_{ij}^+, a_{ij}^-$ with positive coefficients. (Without loss of generality all maxima have the same number of linear terms,

which can always be achieved by repeating linear terms.) We use Corollary B.15 to represent $\mathcal{N}_i$ as a tropical fraction $\frac{\nu_i^+}{\nu_i^-}$ of tropical matrices $\nu_i^+, \nu_i^- \in \mathbb{T}_d$.

Expanding each fraction $\frac{\nu_i^+}{\nu_i^-}$ by $\left(\bigodot_{j\neq i} \nu_j^-\right)$ does not change the evaluation function. Hence, by setting $\mu_i^+ := \nu_i^+ \odot \left(\bigodot_{j\neq i} \nu_j^-\right)$ and $\mu^- := \bigodot_i \nu_i^-$, we obtain a representation $\mathcal{N} = \left(\frac{\mu_1^+}{\mu^-}, \frac{\mu_2^+}{\mu^-}, \ldots, \frac{\mu_n^+}{\mu^-}\right)$. But this is equivalent to $\mathcal{N}_i(\mathbf{x}) = \max\{a_{i1}'^+(\mathbf{x}), \ldots, a_{in}'^+(\mathbf{x})\} - \{a_1'^-(\mathbf{x}), \ldots, a_n'^-(\mathbf{x})\}$ with affine functions $a_{ij}'^+, a_j'^-$ defined by the entries in the tropical matrices $\mu_i^+, \mu^-$.

The label that a network assigns a data point is the argmax of the entries of $\mathcal{N}(\mathbf{x})$. Subtracting the expression on the right in (3) from each entry, leaves the argmax unchanged, so we can discard this expression for classification and only use the vector on the left. In terms of our matrices, the label is then given by

$$\operatorname{argmax}_{1\leq i\leq s} (\mathbf{A}_i^+, \mathbf{a}_i^+)(\mathbf{x}).$$

□

### C.3 Converting Activations Into Their Tropical Form

Let $\sigma : \mathbb{R}^{n_1\times\cdots\times n_t} \to \mathbb{R}^{n_1'\times\cdots\times n_t'}$ be an activation of a neural network. We remind the reader that we denote by $\mathbf{j} = (j_1, \ldots, j_t)$ a multi-index for multi-dimensional arrays. In the following lemmas, we will show how to convert the activations from Table 1 into their tropical forms:

**Lemma C.1** (Leaky ReLU activation)**.** *Let $\sigma : \mathbb{R}^{n_1\times\cdots\times n_t} \to \mathbb{R}^{n_1\times\cdots\times n_t}$ be a Leaky ReLU activation with parameter $0 \leq \alpha < 1$. For all $\mathbf{j} = (j_1, \ldots, j_t)$ with $1 \leq j_k \leq n_k$, let $\mathbf{A}_\mathbf{j}^+ \in \mathbb{R}^{2\times n_1\times\cdots\times n_t}$ contain zeros everywhere except for $(\mathbf{A}_\mathbf{j})_{1,\mathbf{j}}^+ = 1$ and $(\mathbf{A}_\mathbf{j})_{2,\mathbf{j}}^+ = \alpha$. Let $\mathbf{A}_\mathbf{j}^- \in \mathbb{R}^{1\times n_1\times\cdots\times n_t}, \mathbf{a}_\mathbf{j}^+ \in \mathbb{R}^2, \mathbf{a}_\mathbf{j}^- \in \mathbb{R}$ be zero arrays. If $\nu_\mathbf{j}^+ = (\mathbf{A}_\mathbf{j}^+, \mathbf{a}_\mathbf{j}^+)$ and $\nu_\mathbf{j}^- = (\mathbf{A}_\mathbf{j}^-, \mathbf{a}_\mathbf{j}^-)$, then $\nu_\mathbf{j}^+/\nu_\mathbf{j}^-, \in \operatorname{Frac}(\mathbb{T}_d), d = n_1\cdots n_t$ and*

$$\sigma(\mathbf{x}) = \left(\frac{\nu_\mathbf{j}^+}{\nu_\mathbf{j}^-}\right)_\mathbf{j}(\mathbf{x}). \tag{5}$$

**Lemma C.2** (Maxpooling activation)**.** *Let $\sigma : \mathbb{R}^{2n_1\times 2n_2\times n_3} \to \mathbb{R}^{n_1\times n_2\times n_3}$ be a Maxpooling activation with a $2\times 2$ window. For all $\mathbf{j} = (j_1, \ldots, j_t)$ with $1 \leq j_k \leq n_k$, let $\mathbf{A}_\mathbf{j}^+ \in \mathbb{R}^{4\times 2n_1\times 2n_2\times n_3}$ contain zeros everywhere except for $(\mathbf{A}_\mathbf{j})_{1,2j_1,2j_2,j_3}^+ = 1, (\mathbf{A}_\mathbf{j})_{2,2j_1,2j_2+1,j_3}^+ = 1, (\mathbf{A}_\mathbf{j})_{3,2j_1+1,2j_2,j_3}^+ = 1, (\mathbf{A}_\mathbf{j})_{4,2j_1+1,2j_2+1,j_3}^+ = 1$. Let $\mathbf{A}_\mathbf{j}^- \in \mathbb{R}^{1\times n_1\times n_2\times n_3}, \mathbf{a}_\mathbf{j}^+ \in \mathbb{R}^4, \mathbf{a}_\mathbf{j}^- \in \mathbb{R}$ be zero arrays. If $\nu_\mathbf{j}^+ = (\mathbf{A}_\mathbf{j}^+, \mathbf{a}_\mathbf{j}^+)$ and $\nu_\mathbf{j}^- = (\mathbf{A}_\mathbf{j}^-, \mathbf{a}_\mathbf{j}^-)$, then $\nu_\mathbf{j}^+/\nu_\mathbf{j}^-, \in \operatorname{Frac}(\mathbb{T}_d), d = n_1\cdots n_t$ and*

$$\sigma(\mathbf{x}) = \left(\frac{\nu_\mathbf{j}^+}{\nu_\mathbf{j}^-}\right)_\mathbf{j}(\mathbf{x}). \tag{6}$$

*Proofs for Lemmas C.1 and C.2.* These are just simple checks of the definition of tropical evaluation from Definition B.11. □

### C.4 Merging of Tropical Layers

Concatenated layers correspond to the evaluation of fractions of tropical matrices. In this section, we find the corresponding tropical matrices. This will eventually result in Table 1. We begin by considering the concatenation of a tropical rational map with a tropical polynomial.

**Lemma C.3** (Concatenating a tropical rational map with a tropical polynomial)**.** *Let $(\mathbf{B}, \mathbf{b}) \in \mathbb{T}_d$ be the tropical matrix of a tropical polynomial (according to Theorem B.14) and let $\nu = \left(\frac{(\mathbf{A}_\mathbf{j}^+, \mathbf{a}_\mathbf{j}^+)}{(\mathbf{A}_\mathbf{j}^-, \mathbf{a}_\mathbf{j}^-)}\right)_\mathbf{j}$ be the matrix representation of a tropical rational map. Then, using Notation B.17, there is an identity of maps*

$$(\mathbf{B}, \mathbf{b}) \circ \nu = \bigoplus_k \left(\frac{\bigodot_\mathbf{j} (\mathbf{A}_\mathbf{j}^+)^{\mathbf{B}_{k\mathbf{j}}}, \mathbf{b}_k \odot \bigodot_\mathbf{j} (\mathbf{a}_\mathbf{j}^+)^{\mathbf{B}_{k\mathbf{j}}})}{\bigodot_j (\mathbf{A}_\mathbf{j}^-)^{\mathbf{B}_{k\mathbf{j}}}, \mathbf{b}_k \odot \bigodot_\mathbf{j} (\mathbf{a}_\mathbf{j}^-)^{\mathbf{B}_{k\mathbf{j}}})}\right).$$

*Proof.* Let $\mathbf{x}$ be some point in the domain of $\nu$ and let $\mathbf{y} = \nu(\mathbf{x})$. By definition of the evaluation,

$$(\mathbf{B}, \mathbf{b})(\mathbf{y}) = \bigoplus_k \mathbf{b}_k \odot \bigodot_{\mathbf{j}} y_{\mathbf{j}}^{\mathbf{B}_{k\mathbf{j}}} \text{ in } \mathbb{T}.$$

Since $y_{\mathbf{j}} = \left(\frac{(\mathbf{A}_{\mathbf{j}}^+, \mathbf{a}_{\mathbf{j}}^+)}{(\mathbf{A}_{\mathbf{j}}^-, \mathbf{a}_{\mathbf{j}}^-)}\right)(\mathbf{x})$, the result now follows from the fact that for positive $\mathbf{B}_{k\mathbf{j}}$,

$$\left(\left(\frac{(\mathbf{A}_{\mathbf{j}}^+, \mathbf{a}_{\mathbf{j}}^+)}{(\mathbf{A}_{\mathbf{j}}^-, \mathbf{a}_{\mathbf{j}}^-)}\right)(\mathbf{x})\right)^{\mathbf{B}_{k\mathbf{j}}} = \left(\frac{(\mathbf{A}_{\mathbf{j}}^+, \mathbf{a}_{\mathbf{j}}^+)}{(\mathbf{A}_{\mathbf{j}}^-, \mathbf{a}_{\mathbf{j}}^-)}\right)^{\mathbf{B}_{k\mathbf{j}}}(\mathbf{x}).$$

□

Note the necessary condition that $\mathbf{B}_{k\mathbf{j}} \geq 0$ for all $\mathbf{j}$. The following theorems allow us to merge consecutive layers of a neural network to a single representation. Since we aim to sequentially apply these theorems in Algorithm 3.1, we require that the resulting tropical rational function is again represented by an array of fractions of tropical matrices in $\mathbf{T}_d$ satisfying the necessary condition on positivity. Starting from the last layer, this then allows to iteratively merge all layers of the network to obtain a tropical rational function representing the network function. We use the following notation in Theorem C.7:

**Notation C.4.** Given a tropical rational function ${}^{(\mathbf{A}^+, \mathbf{a}^+)}\!/_{(\mathbf{A}^-, \mathbf{a}^-)}$, where $\mathbf{A}^+, \mathbf{A}^- \in \mathbb{R}^{r \times n_1 \times \cdots \times n_t}$, we will denote by $\mathbf{A}_{\max} \in \mathbb{R}^{n_1 \times \cdots \times n_t}$ the array with

$$(\mathbf{A}_{\max})_{\mathbf{j}} = \max\{\mathbf{A}_{1\mathbf{j}}^+, \ldots, \mathbf{A}_{r\mathbf{j}}^+, \mathbf{A}_{1\mathbf{j}}^-, \ldots, \mathbf{A}_{r\mathbf{j}}^-\}.$$

For arrays $\mathbf{v}, \mathbf{w} \in \mathbb{R}^{n_1 \times \cdots \times n_t}$, we will write $\mathbf{v} \leq \mathbf{w}$ if there are entry-wise inequalities $v_{\mathbf{j}} \leq w_{\mathbf{j}}$ for all $\mathbf{j} = (j_1, \ldots, j_t)$.

**Theorem C.5.** *Let $\sigma : \mathbb{R}^{n_1 \times \cdots \times n_t} \to \mathbb{R}^{n_1 \times \cdots \times n_t}$ be a Leaky ReLU activation with parameter $0 \leq \alpha < 1$. Let further $\nu : \mathbb{R}^{n_1 \times \cdots \times n_t} \to \mathbb{R}$ be a tropical rational function represented by a fraction of tropical matrices ${}^{(\mathbf{B}^+, \mathbf{b}^+)}\!/_{(\mathbf{B}^-, \mathbf{b}^-)}$ with $(\mathbf{B}^+, \mathbf{b}^+), (\mathbf{B}^-, \mathbf{b}^-) \in \mathbf{T}_d, d = n_1 \cdots n_t, \mathbf{B}^+, \mathbf{B}^- \in \mathbb{R}^{r \times n_1 \times \cdots \times n_t}$, and $\mathbf{B}_{max} \leq \mathbf{v}$. Let $\mathbf{C}^+ \in \mathbb{R}^{(2^{n_1 \cdots n_t} r) \times n_1 \times \cdots n_t}$ be defined in the following way: For each row $(\mathbf{B}^+)_{k\bullet}$ and each vector $\mathbf{u} \in \{1, \alpha\}^{n_1 \times \cdots \times n_t}$ define a row in $\mathbf{C}_{k'\bullet}^+$ in $\mathbf{C}^+$ by*

$$\mathbf{C}_{k', i_1, \ldots, i_t}^+ := \mathbf{u}_{i_1, \ldots, i_t} \mathbf{B}_{k, i_1, \ldots, i_t}^+.$$

*Define $\mathbf{C}^- \in \mathbb{R}^{(2^{n_1 \cdots n_t} r) \times n_1 \times \cdots n_t}$ in an analogous way, using $\mathbf{B}^-$ instead of $\mathbf{B}^+$. Hence, the rows of the $\mathbf{C}$-arrays are all rows obtained by multiplying elements in rows of the $\mathbf{B}$-arrays by either 1 or $\alpha$. Further, for all $k'$, let $(\mathbf{c}^+)_{k'} = (\mathbf{b}^+)_k$, where $k$ is the row index of $\mathbf{B}^+$ used in definition of $(\mathbf{C}^+)_{k'\bullet}$ (and analogously for $\mathbf{c}^-$). Let $\mathbf{w} = \mathbf{v}$. Then the composition $\nu \circ \sigma$ is represented by the tropical rational function $\frac{(\mathbf{C}^+, \mathbf{c}^+)}{(\mathbf{C}^-, \mathbf{c}^-)}$ with $(\mathbf{C}^+, \mathbf{c}^+), (\mathbf{C}^-, \mathbf{c}^-) \in \mathbf{T}_d$ and $\mathbf{C}_{max} \leq \mathbf{w}$.*

**Theorem C.6.** *Let $\sigma : \mathbb{R}^{2n_1 \times 2n_2 \times n_3} \to \mathbb{R}^{n_1 \times n_2 \times n_3}$ be a Maxpooling activation with a $2 \times 2$ window. Let further $\nu : \mathbb{R}^{n_1 \times n_2 \times n_3} \to \mathbb{R}$ be a tropical rational function represented by a fraction of tropical matrices ${}^{(\mathbf{B}^+, \mathbf{b}^+)}\!/_{(\mathbf{B}^-, \mathbf{b}^-)}$ with $(\mathbf{B}^+, \mathbf{b}^+), (\mathbf{B}^-, \mathbf{b}^-) \in \mathbf{T}_{d_2}, d_2 = n_1 n_2 n_3, \mathbf{B}^+, \mathbf{B}^- \in \mathbb{R}^{r \times n_1 \times n_2 \times n_3}$, and $\mathbf{B}_{max} \leq \mathbf{v}$. Let $\mathbf{C}^+ \in \mathbb{R}^{(4^{n_1 n_2 n_3} r) \times 2n_1 \times 2n_2 \times n_3}$ be defined in the following way: For each row $(\mathbf{B}^+)_{k\bullet}$ and each vector $\mathbf{u} \in \{0, 1\}^{2 \times n_1 \times n_2 \times n_3}$ define a row $\mathbf{C}_{k'\bullet}^+$ in $\mathbf{C}^+$ as zero everywhere except for*

$$\mathbf{C}_{k', (2i_1 + \mathbf{u}_{0,i_1,i_2,i_3}), (2i_2 + \mathbf{u}_{1,i_1,i_2,i_3}), i_3}^+ := \mathbf{B}_{k,i_1,i_2,i_3}^+.$$

*Define $\mathbf{C}^- \in \mathbb{R}^{(4^{n_1 n_2 n_3} r) \times 2n_1 \times 2n_2 \times n_3}$ in an analogous way, using $\mathbf{B}^-$ instead of $\mathbf{B}^+$. Hence, the rows of the $\mathbf{C}$-arrays are all rows obtained by taking a row of the corresponding $\mathbf{B}$-array, repeating its elements by the maxpooling window size and setting 3 of the elements that are covered by the same window to zero. Further, for all $k'$, let $(\mathbf{c}^+)_{k'} = (\mathbf{b}^+)_k$, where $k$ is the row index of $\mathbf{B}^+$ used in the definition of $(\mathbf{C}^+)_{k'\bullet}$ (and analogously for $\mathbf{c}^-$). Let $\mathbf{w} \in \mathbb{R}^{2n_1 \times 2n_2 \times n_3}$ be given by $\mathbf{w}_{2i_1 + \mathbf{u}_0, 2i_2 + \mathbf{u}_1, i_3} = \mathbf{v}_{i_1, i_1, i_3}$ for all $\mathbf{u} \in \{0, 1\}^2$. Then the composition $\nu \circ \sigma$ is represented by the tropical rational function $\frac{(\mathbf{C}^+, \mathbf{c}^+)}{(\mathbf{C}^-, \mathbf{c}^-)}$ with $(\mathbf{C}^+, \mathbf{c}^+), (\mathbf{C}^-, \mathbf{c}^-) \in \mathbf{T}_{d_1}, d_1 = (2n_1)(2n_2)n_3$ and $\mathbf{C}_{max} \leq \mathbf{w}$.*

*Proof.* Since the proofs for both of Theorem C.5 and C.6 proceed in the same fashion, we will only present the first one here and leave the second one to the reader. By symmetry, it suffices to show the result for $\mathbf{C}^+$ and $\mathbf{c}^+$. Lemma C.1 gives us a representation of a Leaky ReLU activation as a tropical rational map $(\nu_{\mathbf{j}}^+/\nu_{\mathbf{j}}^-)_{\mathbf{j}}$ with $\nu_{\mathbf{j}}^+ = (\mathbf{A}_{\mathbf{j}}^+, 0), \nu_{\mathbf{j}}^- = (0,0)$. Applying Lemma C.3, we thus see that

$$(\mathbf{B}^+, \mathbf{b}^+) \circ \nu = \bigoplus_k \left( \bigodot_{\mathbf{j}} (\mathbf{A}_{\mathbf{j}}^+)^{\mathbf{B}_{k\mathbf{j}}^+}, \bigoplus_{\mathbf{j}} \mathbf{b}_k^+ \right).$$

Applying the definition of tropical matrix addition and multiplication shows that this expression is identical to that in Theorem C.5, which proves the result. □

If a layer is linear, we denote its weight matrix by $\mathbf{W} \in \mathbb{R}^{d_2 \times d_1}$ and its bias vector by $\mathbf{c}$. For any matrix $\mathbf{M}$, we denote by $\mathbf{M}^{\text{pos}}$ and $\mathbf{M}^{\text{neg}}$ its positive and negative part, respectively, i.e. $m_{jk}^{\text{pos}} = \max\{m_{jk}, 0\}$ and $m_{jk}^{\text{neg}} = \max\{-m_{jk}, 0\}$.

**Theorem C.7.** *Let $\ell : \mathbb{R}^{d_1} \to \mathbb{R}^{d_2}$ be a linear layer $\ell(\mathbf{x}) = \mathbf{W}\mathbf{x} + \mathbf{c}$ with bias vector $\mathbf{c}$ and weight matrix $\mathbf{W}$. Let further $\nu : \mathbb{R}^{d_2} \to \mathbb{R}$ be a tropical rational function represented by a fraction of tropical matrices $(\mathbf{B}^+, \mathbf{b}^+)/(\mathbf{B}^-, \mathbf{b}^-)$ with $(\mathbf{B}^+, \mathbf{b}^+), (\mathbf{B}^-, \mathbf{b}^-) \in \mathbf{T}_{d_2}$ and let $\mathbf{B}_{max} \leq \mathbf{v}$. Let*

$$\mathbf{K} := \mathbf{v}\mathbf{W}^{neg} \tag{7}$$

$$\mathbf{C}^+ := \mathbf{K} \odot \mathbf{B}^+\mathbf{W}; \qquad \mathbf{c}^+ := \mathbf{B}^+\mathbf{c} + \mathbf{b}^+ \tag{8}$$

$$\mathbf{C}^- := \mathbf{K} \odot \mathbf{B}^-\mathbf{W}; \qquad \mathbf{c}^- := \mathbf{B}^-\mathbf{c} + \mathbf{b}^- \tag{9}$$

$$\mathbf{w} := \mathbf{K} + \mathbf{v}\mathbf{W}^{pos} \tag{10}$$

*Then the composition $\nu \circ \ell$ is represented by the tropical rational function $\frac{(\mathbf{C}^+, \mathbf{c}^+)}{(\mathbf{C}^-, \mathbf{c}^-)}$ with $(\mathbf{C}^+, \mathbf{c}^+), (\mathbf{C}^-, \mathbf{c}^-) \in \mathbf{T}_{d_1}$ and $\mathbf{C}_{max} \leq \mathbf{w}$.*

**Remark C.8.** Before giving the proof, we remark that Theorem C.7 is suitably stated for an inductive use to merge successive layers since the conclusions for $\mathbf{C}^+, \mathbf{C}^-$ satisfy the assumptions on $\mathbf{B}^+, \mathbf{B}^-$. For the same reasons, we considered the vectors $\mathbf{v}$ and $\mathbf{w}$ and $\mathbf{B}_{max}$ in Theorem C.5 and Theorem C.6.

Since convolutional layers are linear functions, the previous theorem applies to their corresponding weight matrices and tropical arrays $\mathbf{B}$ flattened to tropical matrices along all but the first dimension.

*Proof.* By definition of the evaluation,

$$\begin{aligned}
\frac{(\mathbf{C}^+, \mathbf{c}^+)}{(\mathbf{C}^-, \mathbf{c}^-)}(\mathbf{x}) &= \max\{\mathbf{C}^+\mathbf{x} + \mathbf{c}^+\} - \max\{\mathbf{C}^-\mathbf{x} + \mathbf{c}^-\} \\
&= \max\{(\mathbf{K}\mathbf{x} \odot \mathbf{B}^+\mathbf{W}\mathbf{x}) + \mathbf{B}^+\mathbf{c} + \mathbf{b}^+\} - \max\{(\mathbf{K}\mathbf{x} \odot \mathbf{B}^-\mathbf{W}\mathbf{x}) + \mathbf{B}^-\mathbf{c} + \mathbf{b}^-\} \\
&= \mathbf{K}\mathbf{x} + \max\{\mathbf{B}^+\mathbf{W}\mathbf{x} + \mathbf{B}^+\mathbf{c} + \mathbf{b}^+\} - \mathbf{K}\mathbf{x} - \max\{\mathbf{B}^-\mathbf{W}\mathbf{x} + \mathbf{B}^-\mathbf{c} + \mathbf{b}^-\} \\
&= \max\{\mathbf{B}^+(\mathbf{W}\mathbf{x} + \mathbf{c}) + \mathbf{b}^+\} - \max\{\mathbf{B}^-(\mathbf{W}\mathbf{x} + \mathbf{c}) + \mathbf{b}^-\} \\
&= \frac{(\mathbf{B}^+, \mathbf{b}^+)}{(\mathbf{B}-, \mathbf{b}^-)}(\ell(\mathbf{x})),
\end{aligned}$$

which shows that the composition $\nu \circ \ell$ is represented by $\frac{(\mathbf{C}^+, \mathbf{c}^+)}{(\mathbf{C}^-, \mathbf{c}^-)}$.

We now need to show that $\mathbf{K} \in \mathbb{R}^{1 \times d_1}$ is a sufficiently large row vector such that all entries in $\mathbf{C}^+$ and $\mathbf{C}^-$ are positive and hence give indeed elements in $\mathbf{T}_d$. For this, let $\mathbf{K}_{\text{min}}$ denote the vector with minimal entries for which the matrices in (8) and (9) have only non-negative entries. We need to show that $\mathbf{K}_{\text{min}} \leq \mathbf{K}$:

Since the minimal element of the $i^{\text{th}}$ columns of $\mathbf{C}^+$ and $\mathbf{C}^-$ are given by $\min_k \sum_j \mathbf{B}_{kj}^+ w_{ji}$ and $\min_k \sum_j \mathbf{B}_{kj}^- w_{ji}$, respectively, it follows that

$$\mathbf{K}_{min,i} = \max\{-\min_k \sum_j \mathbf{B}_{kj}^+ w_{ji}, -\min_l \sum_j \mathbf{B}_{lj}^- w_{ji}\}.$$

With

$$w_{ji}^{\text{neg}} = \begin{cases} -w_{ji}, & \text{if } w_{ji} < 0 \\ 0, & \text{otherwise} \end{cases}$$

we calculate

$$\begin{aligned} -\min_k \sum_j \mathbf{B}^+_{kj} w_{ji} &= \max_k \sum_j \mathbf{B}^+_{kj}(-w_{ji}) \\ &\leq \max_k \sum_j \mathbf{B}^+_{kj} w^{\text{neg}}_{ji} \\ &\leq \sum_j (\max_k \mathbf{B}^+_{kj}) w^{\text{neg}}_{ji} \\ &\leq \sum_j \mathbf{v}_j w^{\text{neg}}_{ji} \\ &= \mathbf{v} \cdot \mathbf{W}^{\text{neg}}_{\bullet i}. \end{aligned}$$

Analogously, we get $-\min_l \sum_j \mathbf{B}^-_{lj} w_{ji} \leq \mathbf{v} \cdot \mathbf{W}^{\text{neg}}_{\bullet i}$ and hence $\mathbf{K}_{min,i} \leq \mathbf{v} \cdot \mathbf{W}^{\text{neg}}_{\bullet i}$ for all $i$.

It only remains to show that $\mathbf{C}_{max} \leq \mathbf{w}$. The maximal element of the $i^{\text{th}}$ columns of $\mathbf{C}^+$ and $\mathbf{C}^-$ for $i \geq 1$ is given by

$$\max \left\{ K_i + \max_k \sum_j \mathbf{B}^+_{kj} w_{ji},\ K_i + \max_l \sum_j \mathbf{B}^-_{lj} w_{ji} \right\}.$$

With

$$w^{\text{pos}}_{ji} = \begin{cases} w_{ji}, & w_{ji} > 0 \\ 0, & \text{otherwise} \end{cases},$$

we calculate

$$\begin{aligned} K_i + \max_k \left( \sum_j \mathbf{B}^+_{kj} w_{ji} \right) &\leq K_i + \max_k \left( \sum_j \mathbf{B}^+_{kj} w^{\text{pos}}_{ji} \right) \\ &\leq K_i + \sum_j (\max_k \mathbf{B}^+_{kj}) w^{\text{pos}}_{ji} \\ &\leq K_i + \sum_j \mathbf{v}_j w^{\text{pos}}_{ji} \\ &= K_i + \mathbf{v}_j \mathbf{W}^{\text{pos}}_{\bullet i}. \end{aligned}$$

Analogously

$$K_i + \max_l \left( \sum_j \mathbf{B}^-_{lj} w_{ji} \right) \leq K_i + \mathbf{B}_{\text{max}} \mathbf{W}^{\text{pos}}_{\bullet i}$$

and hence $\mathbf{C}_{max} \leq \mathbf{K} + \mathbf{B}_{\text{max}} \mathbf{W}^{\text{pos}} = \mathbf{w}$, which completes the proof. □

### C.5 Selection of Terms

To prevent the tropical matrices that describe concatenations of several layers to explode in size, we want to select linear terms during the merge operation. The evaluation of a tropical matrix at a point $\mathbf{x}$ (Definition B.11) searches through its rows for the one that is maximal on $\mathbf{x}$. We give the maximizing rows names:

**Notation C.9.** For any tropical matrix $\nu = (\mathbf{A}, \mathbf{a})$ and data point $\mathbf{x}$, let us denote by $[\mathbf{A}]_{\mathbf{x}}$ and $[\mathbf{a}]_{\mathbf{x}}$ the rows of $\mathbf{A}$ and $\mathbf{a}$, respectively, that correspond to the maximal row in $\mathbf{A}_{\text{flat}}\mathbf{x}_{\text{flat}} + \mathbf{a}$.

Suppose that we want to use Theorems C.5, C.6 and C.7 to merge a tropical rational function $(\mathbf{B}^+, \mathbf{b}^+)/(\mathbf{B}^-, \mathbf{b}^-)$ with a linear layer $\ell$ or an activation $\sigma$ to obtain $(\mathbf{C}^+, \mathbf{c}^+)/(\mathbf{C}^-, \mathbf{c}^-)$. We will show that the rows of the resulting arrays maximal on $\mathbf{x}$ are completely determined by the layer or the activation and by the rows of the $\mathbf{B}$-arrays maximal on $\ell(\mathbf{x})$ or $\sigma(\mathbf{x})$.

**Theorem C.10.** *Let $\sigma : \mathbb{R}^{n_1 \times \cdots \times n_t} \to \mathbb{R}^{n_1 \times \cdots \times n_t}$ be a Leaky ReLU activation with parameter $0 \leq \alpha < 1$. Let $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ denote a data set of $N$ points before layer $\ell$ and let $T = \ell(S)$ be the set of points having passed through $\ell$. Let further $\nu : \mathbb{R}^{n_1 \times \cdots \times n_t} \to \mathbb{R}$ be a tropical rational*

*function represented by a fraction of tropical matrices* $(\mathbf{B}^+, \mathbf{b}^+)/(\mathbf{B}^-, \mathbf{b}^-)$ *with* $(\mathbf{B}^+, \mathbf{b}^+), (\mathbf{B}^-, \mathbf{b}^-) \in \mathbf{T}_d$*, with* $\mathbf{B}^+, \mathbf{B}^- \in \mathbb{R}^{r \times n_1 \times \cdots \times n_t}$*, and* $d = n_1 n_2 \dots n_t$*. Let us suppose that the* $n^{th}$ *rows of* $\mathbf{B}^+, \mathbf{b}^+, \mathbf{B}^-, \mathbf{b}^-$ *are maximal on* $\mathbf{y}_n = \ell(\mathbf{x}_n) \in T$*.*

*Let* $(\mathbf{C}^+, \mathbf{c}^+)/(\mathbf{C}^-, \mathbf{c}^-)$ *be given by Theorem C.5 and as before let* $\mathbf{j} = (j_1, \dots, j_t)$ *denote a multi-index for* $1 \leq j_i \leq n_i$*. Then*

$$\left([\mathbf{C}^+]_{\mathbf{x}_n}\right)_{\mathbf{j}} = \begin{cases} ([\mathbf{B}^+]_{\mathbf{x}_n})_{\mathbf{j}} & \text{if } (\ell(\mathbf{x}_n))_{\mathbf{j}} \geq 0; \\ \alpha\,([\mathbf{B}^+]_{\mathbf{x}_n})_{\mathbf{j}} & \text{otherwise}; \end{cases} \tag{11}$$

$$\left([\mathbf{C}^-]_{\mathbf{x}_n}\right)_{\mathbf{j}} = \begin{cases} ([\mathbf{B}^-]_{\mathbf{x}_n})_{\mathbf{j}} & \text{if } (\ell(\mathbf{x}_n))_{\mathbf{j}} \geq 0; \\ \alpha\,([\mathbf{B}^-]_{\mathbf{x}_n})_{\mathbf{j}} & \text{otherwise}. \end{cases} \tag{12}$$

*Further,* $[\mathbf{c}^+]_{\mathbf{x}_n} = (\mathbf{b}^+)_n$ *and* $[\mathbf{c}^-]_{\mathbf{x}_n} = (\mathbf{b}^-)_n$*. In particular, if we re-order the rows of the* $\mathbf{C}$*-arrays and* $\mathbf{c}$*-vectors such that* $(\mathbf{C}^+)_{n\bullet} := [\mathbf{C}^+]_{\mathbf{x}_n}, (\mathbf{C}^-)_{n\bullet} := [\mathbf{C}^-]_{\mathbf{x}_n}$ *and* $(\mathbf{c}^+)_{n\bullet} := [\mathbf{c}^+]_{\mathbf{x}_n}, (\mathbf{c}^-)_{n\bullet} := [\mathbf{c}^-]_{\mathbf{x}_n}$*, then their* $n^{th}$ *rows are maximal on* $\mathbf{x}_n \in S$*.*

**Theorem C.11.** *Let* $\sigma : \mathbb{R}^{2n_1 \times 2n_2 \times n_3} \to \mathbb{R}^{n_1 \times n_2 \times n_3}$ *be a Maxpooling activation. Let* $S = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ *denote a data set of* $N$ *points before layer* $\ell$ *and let* $T = \ell(S)$ *be the set of points having passed through* $\ell$*. Let further* $\nu : \mathbb{R}^{n_1 \times n_2 \times n_3} \to \mathbb{R}$ *be a tropical rational function represented by a fraction of positive tropical matrices* $(\mathbf{B}^+, \mathbf{b}^+)/(\mathbf{B}^-, \mathbf{b}^-)$ *with* $(\mathbf{B}^+, \mathbf{b}^+), (\mathbf{B}^-, \mathbf{b}^-) \in \mathbf{T}_d, d = n_1 n_2 n_3$*. Let us suppose that the* $n^{th}$ *rows of* $\mathbf{B}^+, \mathbf{b}^+, \mathbf{B}^-, \mathbf{b}^-$ *are maximal on* $\mathbf{y}_n = \ell(\mathbf{x}_n) \in T$*.*

*Let* $(\mathbf{C}^+, \mathbf{c}^+)/(\mathbf{C}^-, \mathbf{c}^-)$ *be given by Theorem C.6. Then for all* $\mathbf{u} \in \{0, 1\}^2$*,*

$$\left([\mathbf{C}^+]_{\mathbf{x}_n}\right)_{(2i_1+\mathbf{u}_0),(2i_2+\mathbf{u}_1),i_3} = \begin{cases} ([\mathbf{B}^+]_{\mathbf{x}_n})_{i_1,i_2,i_3} & \text{if } (\mathbf{x}_n)_{(2i_1+\mathbf{u}_0),(2i_2+\mathbf{u}_1),i_3} = (\ell(\mathbf{x}_n))_{i_1,i_2,i_3}; \\ 0 & \text{otherwise}; \end{cases} \tag{13}$$

$$\left([\mathbf{C}^-]_{\mathbf{x}_n}\right)_{(2i_1+\mathbf{u}_0),(2i_2+\mathbf{u}_1),i_3} = \begin{cases} ([\mathbf{B}^-]_{\mathbf{x}_n})_{i_1,i_2,i_3} & \text{if } (\mathbf{x}_n)_{(2i_1+\mathbf{u}_0),(2i_2+\mathbf{u}_1),i_3} = (\ell(\mathbf{x}_n))_{i_1,i_2,i_3}; \\ 0 & \text{otherwise}. \end{cases} \tag{14}$$

*Further,* $[\mathbf{c}^+]_{\mathbf{x}_n} = (\mathbf{b}^+)_n$ *and* $[\mathbf{c}^-]_{\mathbf{x}_n} = (\mathbf{b}^-)_n$*. In particular, if we re-order the rows of the* $\mathbf{C}$*-arrays and* $\mathbf{c}$*-vectors such that* $(\mathbf{C}^+)_{n\bullet} := [\mathbf{C}^+]_{\mathbf{x}_n}, (\mathbf{C}^-)_{n\bullet} := [\mathbf{C}^-]_{\mathbf{x}_n}$ *and* $(\mathbf{c}^+)_{n\bullet} := [\mathbf{c}^+]_{\mathbf{x}_n}, (\mathbf{c}^-)_{n\bullet} := [\mathbf{c}^-]_{\mathbf{x}_n}$*, then their* $n^{th}$ *rows are maximal on* $\mathbf{x}_n \in S$*.*

*Proof.* Again, since the proofs for both of Theorem C.10 and C.11 apply almost the same argument, we will only present the one for Leaky ReLU activations. Once more, it is enough to prove the result for $\mathbf{C}^+$ and $\mathbf{c}^+$ due to symmetry. Let us fix some data point $\mathbf{x}_n \in S$ and let us denote the corresponding row defined in equation (11) by $(\mathbf{C}^+)^*$. We will show that this row is indeed maximal on $\mathbf{x}_n$:

$$\begin{aligned} (\mathbf{C}^+)^*\mathbf{x}_n + (\mathbf{b}^+)_n &= \sum_{\mathbf{j}} (\mathbf{C}^+)^*_{\mathbf{j}} (\mathbf{x}_n)_{\mathbf{j}} + (\mathbf{b}^+)_n \\ &\geq \sum_{\mathbf{j}} \mathbf{u}_{\mathbf{j}} (\mathbf{B}^+)_{n\mathbf{j}} (\mathbf{x}_n)_{\mathbf{j}} + (\mathbf{b}^+)_n \\ &\geq \sum_{\mathbf{j}} \mathbf{u}_{\mathbf{j}} (\mathbf{B}^+)_{n'\mathbf{j}} (\mathbf{x}_n)_{\mathbf{j}} + (\mathbf{b}^+)_{n'}, \end{aligned}$$

for any $\mathbf{u} \in \{1, \alpha\}^{n_1 \times \cdots \times n_t}$. Here, the first inequality holds because the first line gets maximized when $\mathbf{u}$ agrees with the activation patterns of $\mathbf{x}_n$. The second inquality holds by assumption on $\mathbf{B}^+$. Since the evaluation of $\mathbf{x}_n$ on any row of $\mathbf{C}^+$ takes the same form as the last expression, it follows that $\mathbf{x}_n$ is maximal on $(\mathbf{C}^+)^*$ and $(\mathbf{b}^+)_n$. This proves the result. □

The following theorem extracts the linear terms from the representation of the network function constructed by Theorem C.7.

**Theorem C.12.** *Let* $\ell : \mathbb{R}^{d_1} \to \mathbb{R}^{d_2}$ *be a linear layer,* $\ell(\mathbf{x}) = \mathbf{W}\mathbf{x} + \mathbf{c}$ *with weight matrix* $\mathbf{W}$ *and bias vector* $\mathbf{c}$*. Let* $S = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ *denote a data set of* $N$ *points before layer* $\ell$ *and let* $T = \ell(S)$

*be the set of points having passed through $\ell$. Let further $\nu : \mathbb{R}^{d_2} \to \mathbb{R}$ be a tropical rational function represented by a fraction of tropical matrices $\frac{\mathbf{B}^+}{\mathbf{B}^-}$ with $\mathbf{B}^+, \mathbf{B}^- \in \mathbf{T}_{d_2}$. Let us suppose that the $n^{th}$ rows of $\mathbf{B}^+, \mathbf{B}^-$ are maximal on $\mathbf{y}_n = \ell(\mathbf{x}_n) \in T$.*

*Let $\mathbf{C}^+$ and $\mathbf{C}^-$ be given by the expressions* (8) *and* (9) *from Theorem C.7. Then*

$$\left[\mathbf{C}^+\right]_{\mathbf{x}_n} = \mathbf{K} + \left(\mathbf{B}^+\right)_{n\bullet}\mathbf{W}, \qquad \left[\mathbf{c}^+\right]_{\mathbf{x}_n} = \left(\mathbf{B}^+\right)_{n\bullet}\mathbf{c} + (\mathbf{b}^+)_n, \tag{15}$$

$$\left[\mathbf{C}^-\right]_{\mathbf{x}_n} = \mathbf{K} + \left(\mathbf{B}^-\right)_{n\bullet}\mathbf{W}, \qquad \left[\mathbf{c}^-\right]_{\mathbf{x}_n} = \left(\mathbf{B}^-\right)_{n\bullet}\mathbf{c} + (\mathbf{b}^-)_n. \tag{16}$$

*In particular, if we re-order the rows of the $\mathbf{C}$-matrices such that $(\mathbf{C}^+)_{n\bullet} := \left[\mathbf{C}^+\right]_{\mathbf{x}_n}$ and $(\mathbf{C}^-)_{n\bullet} := \left[\mathbf{C}^-\right]_{\mathbf{x}_n}$, then their $n^{th}$ rows are maximal on $\mathbf{x}_n \in S$.*

*Proof.* It suffices to show that the rows of $(\mathbf{C}^+, \mathbf{c}^+)$ and $(\mathbf{C}^-, \mathbf{c}^-)$ defined as in 15 and 16 are indeed maximal on $\mathbf{x}_n$:

$$\begin{aligned}
(\mathbf{K} + (\mathbf{B}^+)_{n\bullet}\mathbf{W})\mathbf{x}_n + (\mathbf{B}^+)_{n\bullet}\mathbf{c} + (\mathbf{b}^+)_n &= \mathbf{K}\mathbf{x}_n + (\mathbf{B}^+)_{n\bullet}(\mathbf{W}\mathbf{x}_n + \mathbf{c}) + (\mathbf{b}^+)_n \\
&= \mathbf{K}\mathbf{x}_n + (\mathbf{B}^+)_{n\bullet}\mathbf{y}_n + (\mathbf{b}^+)_n \\
&\geq \mathbf{K}\mathbf{x}_n + (\mathbf{B}^+)_{n'\bullet}\mathbf{y}_n + (\mathbf{b}^+)_{n'} \\
&= (\mathbf{K} + (\mathbf{B}^+)_{n'\bullet}\mathbf{W})\mathbf{x}_n + (\mathbf{B}^+)_{n'\bullet}\mathbf{c} + (\mathbf{b}^+)_{n'}
\end{aligned}$$

for any number $n'$, since $\mathbf{B}^+$ and $\mathbf{b}^+$ are maximal on $\mathbf{x}_n$. Arguing analogously for $\mathbf{C}^-$ and $\mathbf{c}^-$ proves the result. □

### C.6 Converting the Last Layer Into a Tropical Rational Function

**Lemma C.13.** *Let $\ell : \mathbb{R}^{d_{L-1}} \to \mathbb{R}^s$, $\ell(\mathbf{x}) = \mathbf{W}\mathbf{x}+\mathbf{b}$ be the last linear layer of a ReLU classification neural network $\mathcal{N} = (\mathcal{N}_1, \ldots, \mathcal{N}_s) : \mathbb{R}^d \to \mathbb{R}^s$ into $s$ classes. (If the layer contains a softmax activation function, we can leave it out because it does not change the ordering of the network outputs.) Let $\nu = \left((\mathbf{A}_i^+, \mathbf{a}_i^+)/(\mathbf{A}^-, \mathbf{a}^-)\right)_{1\leq i\leq s}$ be the tropical rational function obtained by lines 1-7 of Algorithm 3.1. Then $\nu(\mathbf{x}) = \ell(\mathbf{x})$ for all $\mathbf{x} \in \mathbb{R}^{d_{L-1}}$.*

*Proof.* We have

$$\begin{aligned}
\nu_i(\mathbf{x}) &= \max\{\mathbf{A}_i^+\mathbf{x} + \mathbf{a}_i^+\} - \max\{\mathbf{A}^-\mathbf{x} + \mathbf{a}^+\} \\
&= (\mathbf{C}_i^+\mathbf{x} + \mathbf{c}_i^+) - (\mathbf{C}^-\mathbf{x} + \mathbf{c}^-) \\
&= ((\mathbf{W}_i + \mathbf{C}^-)^+\mathbf{x} + \mathbf{b}_i + \mathbf{c}^-) - (\mathbf{C}^-\mathbf{x} + \mathbf{c}^-) \\
&= \mathbf{W}_i\mathbf{x} + \mathbf{b}_i \\
&= \ell_i(\mathbf{x}),
\end{aligned}$$

as required. □

### C.7 Proof of Theorem 3.3

Let $\mathcal{N} = (\mathcal{N}_1, \ldots, \mathcal{N}_s) : \mathbb{R}^d \to \mathbb{R}^s$ be the function of a ReLU neural network for classification into $s$ classes. Let $\mathcal{N}^{(\mathcal{X})}$ be the network obtained from Algorithm 3.1, applied to $\mathcal{N}$ using a data set $\mathcal{X} = \{(\mathbf{x}_{i_k}, i)\}_{1\leq i\leq s}$ of $D_i$ data points $\mathbf{x}_{i_k}$ given label $i$ by $\mathcal{N}$. (There are $D$ points in total).

We want to show (1) for all labels $i$, $\mathcal{N}_i^{(\mathcal{X})}(\mathbf{x}) = \max\{a_{i1}^+(\mathbf{x}), \ldots, a_{iD_i}^+(\mathbf{x})\} - \max\{a_1^-(\mathbf{x}), \ldots, a_D^-(\mathbf{x})\}$, where the $a_{ij}^+$ and $a_j^-$ are extracted from a representation of $\mathcal{N}_i$ as in Equation 3; and (2) every data point $\mathbf{x}_k$ has a neighbourhood $U_k$ on which the maximum of the extracted function agrees with the maximal network output:

$$\max_{1\leq i\leq s} \mathcal{N}_i^{(\mathcal{X})}(\mathbf{x}) = \max_{1\leq i\leq s} \mathcal{N}_i(\mathbf{x}) \quad \text{for all } \mathbf{x} \in U_k.$$

*Proof.* Using the correspondence of tropical rational functions and $\mathrm{Frac}(\mathbb{T}_d)$ from Section B it suffices to find a representation of $\mathcal{N}_i = \frac{(\mathbf{B}_i^+, \mathbf{b}_i^+)}{(\mathbf{B}^-, \mathbf{b}^-)}$ with tropical matrices $(\mathbf{B}_i^+, \mathbf{b}_i^+), (\mathbf{B}^-, \mathbf{b}^-) \in \mathbf{T}_d$ and submatrices $(\mathbf{A}_i^+, \mathbf{a}_i^+), (\mathbf{A}^-, \mathbf{a}^-)$ of $(\mathbf{B}_i^+, \mathbf{b}_i^+), (\mathbf{B}^-, \mathbf{b}^-)$ respectively by selection of rows based on the data set $\mathcal{X}$. The tropical matrices $(\mathbf{A}_i^+, \mathbf{a}_i^+), (\mathbf{A}^-, \mathbf{a}^-) \in \mathbf{T}_d$ then define $\nu$ with the properties as desired.

Lemma C.13 treats the last layer, which is implemented by lines 1-7 of Algorithm 3.1. We can use Section C.4 to successively merge layers until the whole network function is encoded in tropical matrices $(\mathbf{B}_i^+, \mathbf{b}_i^+), (\mathbf{B}^-, \mathbf{b}^-)$ as desired. Theorem C.12 shows how to calculate the selected linear terms for dense layers to yield tropical matrices $(\mathbf{A}_i^+, \mathbf{a}_i^+), (\mathbf{A}^-, \mathbf{a}^-)$ as desired. The results for merging (leaky) ReLU and maxpooling layers follow from Theorems C.10 and C.11. Since batch normalization and convolutions are linear functions, we can treat them similarly to dense layers. In particular, if $F$ is the filter of a convolutional layer and $\mathbf{W}_F$ the matrix of the corresponding linear function, we have $\mathbf{A}\mathbf{W}_F = (\mathbf{W}_F^T \mathbf{A}^T)^T = \text{ConvTrans}(\mathbf{A}^T, F)^T = \text{ConvTrans}(\mathbf{A}, F)$, where ConvTrans denotes transposed convolution. This shows that Table 1 performs the equations required by the theorems in Section C.4 and Section C.5. □

### C.8 Using Different Values of $K$

In Theorem C.7 we derive a value of $K$ which is large enough to guarantee that the resulting $\mathbf{C}$-matrices are all positive. Algorithm 3.1 uses a slightly different $K$. Since we extract rows while merging layers, the resulting tropical matrices have much less rows than the full network would have. As a consequence, calculating $K$ only from the selected rows may lead to smaller values only assuring that the extracted rows are all positive. We therefore experimented with larger values of $K$, which were just large enough to guarantee that all elements of the $\mathbf{A}$-matrices were positive and did not notice any significant difference in the performance of the resulting function $\mathcal{N}^{(\mathcal{X})}$. However, for smaller value of $K$, we do not have a theoretical guarantee that the linear terms in $\mathcal{N}^{(\mathcal{X})}$ stem indeed from a representation of $\mathcal{N}$, so we always used the large value in the reported experiments.

### C.9 An Observation on Cross Terms

As a result of our algorithm, we have $\nu(\mathbf{x}_i) = p_i(\mathbf{x}_i) - q_i(\mathbf{x}_i)$ for all training samples, i.e., $\mathbf{x}_i$ maximizes the $i^{\text{th}}$ function in both terms. Other samples $\mathbf{x}$ can in theory maximize functions with differing indices, $\nu(\mathbf{x}) = p_i(\mathbf{x}) - q_j(\mathbf{x})$ for $i \neq j$, and the extracted function $\nu$ can have up to $N^2$ different linear regions, although a much lower number can be expected. Further, the combination of maximizing functions $p_i, q_j$ can be such that the same combination is never possible for the original network, which could have a serious impact on our results. However, we observe that for all but the small architectures, $p_i$ is maximal on a test data point if and only if $q_i$ is maximal on it. Thus we observe only linear functions that are also seen on training data points and the extraction process does not create any new linear function $p_i - q_j$ on the test data that was previously unseen.

## D SETUP OF EXPERIMENTS

All layers use biases and He uniform variance scaling initialisation (He et al., 2015). The "big deep simple" architecture FCN6 on MNIST is taken from (Claudiu Ciresan et al., 2010). The architecture *Conv* is a slightly modified VGG-network (Simonyan & Zisserman, 2015). We employ convolutional filters of sizes between $1 \times 1$ and $7 \times 7$. All convolutional layers use zero padding to leave the image size unchanged. All maxpooling layers use a $2 \times 2$ window. All stride layers have a stride of $2 \times 2$. The convolutional CIFAR-networks are trained using stochastic gradient descent, all the other networks are trained using Adam (Kingma & Ba, 2015). We train AllCNN for a total of 350 epochs with an initial learning rate of 0.1 that decays by a factor of 0.1 after 200, 250 and 300 epochs. All the other networks are trained with a learning rate of 0.001 for a maximum of 50 epochs, or if their accuracy on the test set has stopped improving for 3 epochs. An overview of all network structures is given in Table 3. For MNIST, the architectures comprise one 4-layer and one 6-layer fully-connected network, as well as one 4-layer convolutional network. For CIFAR, we use 2-layer, 4-layer and 8-layer fully-connected networks and 8-layer and 9-layer CNNs with/without batchnorm, maxpooling after convolutions, and dropout (Hinton et al., 2012).

| MNIST | | | CIFAR10 | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **FCN4** | **FCN6** | **CNN - Std** | **FCN2** | **FCN4** | **FCN8** | **Wide** | **AllCNN** | **Conv** | **Deep** | **Narrow** | **NarrowStride** | **NoMax** |
| D784 | D2500 | C32+M | D3072 | D3072 | D4096 | D4096 | C96+O2 | C32 | C32-7 | C4 | C4 | C8 |
| D200 | D2000 | C64 | S10 | D400 | D3072 | D4096 | C96 | C32+M | C64-5+M | C4+M | C4+St | C16 |
| D80 | D1500 | C64+M | | D120 | D2048 | D4096 | C64+O5 | C64 | C128 | C16 | C16 | C16 |
| S10 | D1000 | D64 | | S10 | D1024 | D4096 | C192 | C64+M | C256+M | C16+M | C16+St | C32 |
| | D500 | S10 | | | D512 | D4096 | C192 | C128 | C512 | C64 | C64 | C32 |
| | S10 | | | | D256 | D4096 | C192+O5 | C128+M | C1024 | C64+M | C64+St | C64 |
| | | | | | D128 | D128 | C192-1 | D128 | D1024 | D128 | D128 | D128 |
| | | | | | S10 | S10 | C10-1 | S10 | D128 | S10 | S10 | S10 |
| | | | | | | | GA+S10 | | S10 | | | |

Table 3: Network structures used for training. There is a ReLU activation after every dense or convolutional layer. C - Convolutional; D- Dense; GA - Global Average Pooling; M - Maxpooling; O - Dropout; S - Softmax; St - Strides. Numbers after layer type indicate: No of output nodes (D,S); no of output channels (C); dropout parameter multiplied by 10 (O). Standard convolutional filter size is 3x3, deviations are indicated by number after the sign - (i.e. 7x7 filter in C32-7). All convolutional layers in **Deep** are repeated twice, which is omitted due to lack of space (i.e. C512 in column **Deep** should be read as C512, C512). Additional network: **BaNorm** is identical to **Conv** with batch normalization added after every convolutional layer.

# E ADDITIONAL EXPERIMENTS

## E.1 ACCURACY OF EXTRACTED FUNCTIONS - INDIVIDUAL RESULT FOR EACH ARCHITECTURE

To investigate how well the coefficients of these linear regions generalize to unseen data, we compare test accuracy of network and extracted function in Table 2. In Table 4 we show the results for each individual network structure. The difference between CNNs and FCNs across both data sets and all architectures is indeed consistent: There is a drastic drop in the test accuracy of the CNNs, as opposed to a relatively small drop in the accuracy of the FCNs. We repeat that, as predicted by Theorem 3.3, the maximum of the extracted function $\mathcal{N}^{(\mathcal{X})}$ agrees with the maximum of the original network on all training points for each of our networks. In particular, network and extracted function assign the same label to each training point.

| | Name | $\mathcal{N}$ vs $\nu$ | $\mathcal{N}$ vs true | $\nu$ vs true |
|---|---|---|---|---|
| MNIST | **FCN4** | $97.7 \pm 0.1$ | $98.1 \pm 0.1$ | $97.7 \pm 0.1$ |
| | **FCN6** | $97.7 \pm 0.3$ | $98.1 \pm 0.2$ | $97.5 \pm 0.2$ |
| | **CNN - Std** | $95.7 \pm 0.3$ | $99.2 \pm 0.1$ | $95.8 \pm 0.2$ |
| CIFAR | **FCN2** | $74.4 \pm 3.7$ | $49.1 \pm 0.8$ | $46.0 \pm 0.4$ |
| | **FCN4** | $52.3 \pm 1.5$ | $51.8 \pm 0.7$ | $40.4 \pm 0.8$ |
| | **FCN8** | $47.3 \pm 4.4$ | $49.0 \pm 0.4$ | $35.5 \pm 1.9$ |
| | **Wide** | $57.8 \pm 4.6$ | $46.7 \pm 1.0$ | $38.6 \pm 1.7$ |
| | **AllCNN** | $32.6 \pm 0.1$ | $86.9 \pm 0.1$ | $32.4 \pm 0.1$ |
| | **BaNorm** | $27.6 \pm 0.1$ | $70.2 \pm 0.1$ | $27.7 \pm 0.1$ |
| | **Conv** | $29.4 \pm 0.9$ | $71.8 \pm 0.7$ | $28.6 \pm 1.2$ |
| | **Deep** | $37.7 \pm 0.6$ | $71.2 \pm 0.5$ | $38.0 \pm 0.8$ |
| | **Narrow** | $27.5 \pm 3.1$ | $63.1 \pm 0.6$ | $26.5 \pm 2.9$ |
| | **NarrowStride** | $32.7 \pm 1.6$ | $74.4 \pm 2.6$ | $32.6 \pm 2.0$ |
| | **NoMax** | $29.9 \pm 2.8$ | $63.2 \pm 0.9$ | $28.4 \pm 3.4$ |
| Fashion MNIST | **FCN4** | $88.3 \pm 0.3$ | $85.2 \pm 0.6$ | $82.9 \pm 0.4$ |
| | **FCN6** | $88.1 \pm 0.1$ | $85.5 \pm 0.9$ | $83.5 \pm 0.1$ |
| | **FCN8** | $88.5 \pm 0.2$ | $86.0 \pm 0.8$ | $84.0 \pm 0.6$ |
| | **AllCNN** | $87.1 \pm 1.0$ | $76.2 \pm 9.7$ | $73.5 \pm 0.8$ |
| | **CNN - Std** | $86.3 \pm 0.6$ | $89.0 \pm 1.9$ | $83.5 \pm 1.4$ |
| | **Conv** | $87.5 \pm 0.6$ | $87.5 \pm 0.8$ | $87.8 \pm 0.5$ |

Table 4: Results on test data after extraction of linear terms. Values denote mean value and standard deviation over 5 networks in percent. Column 1: Agreement of original network $\mathcal{N}$ with reduced function $\nu$. Columns 2&3: Multi-class accuracy for $\mathcal{N}$ and $\nu$.

## E.2 VISUAL EXPLORATION OF GENERALIZATION

We visually inspect samples that are correctly classified by a network $\mathcal{N}$, but misclassified after extraction by $\mathcal{N}^{(\mathcal{X})}$, i.e., samples that the original network is able to correctly generalize to, but for which the reduction of linear regions is hurtful. Figure 8 shows five random examples for both a CNN and an FCN across the first five CIFAR10-labels *1:airplane, 2:automobile, 3:bird, 4:cat, 5:deer*, with the first row showing the test samples and the second row the training images corresponding to the linear regions in $\mathcal{N}^{(\mathcal{X})}$ falsely activated after extraction. Note that by extracting linear terms we remove the compositional structure of the network function, and the extracted function must rely on the information encoded in the linear coefficients of linear regions containing training data. Inspecting samples that were correctly classified by the original network, but misclassified by the extraction, can give us insight what generalization capabilites were removed by extraction that is not contained in the linear coefficients. We see that for CNNs, the generalization performance of the extracted function seems to rely on color distribution and background. This supports our interpretation of in Section 4 predicting that removal of structure by extraction leads to increased vulnerability to spurious correlations for CNNs. In contrast, for FCNs, the extracted functions appears to classify images more according to the global shape present.

Figure 8: First row: Examples of test images correctly classified by the original network, but misclassified by the tropical function, Second row: Training images corresponding to the linear region falsely activated by the test sample. Left: CNN (Conv). Right: Fully-Connected Network (FCN8).

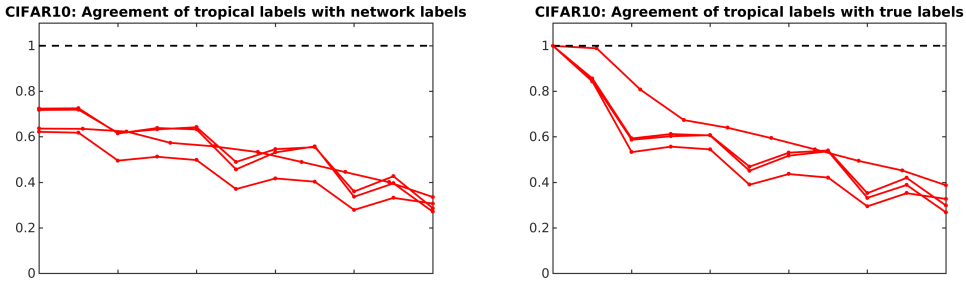



Figure 9: Average agreement with the labels computed by the neural network (left) and average test accuracy (right) for several CNNs with LeakyReLU activations on CIFAR10 while being transformed into a tropical function. The convolutional networks are shown as full red (light) lines. Our results are in line with our observations for standard ReLU activations.

### E.3 Leaky ReLU

The results of our additional experiments involving Leaky ReLU activation (with slope $0.3$ for negative input) can be found in Figure 9, confirming that the same observations hold for this activation function.

### E.4 Evolution of test accuracy during extraction

We show additional plots in Figure 10 and 11 on the comparison of fully-connected and convolutional networks. We show here for CIFAR10 the agreement of network labels and for MNIST both agreement with network labels as well as agreement with true test labels.

The results agree with the results presented in the main part of the paper: When extracting linear regions from a ReLU neural network, we experience a very small drop in test accuracy for fully-connected networks and a substantial drop for convolutional neural networks.

### E.5 Similarity of activation patterns

By counting and comparing linear regions, we found that consistently all training and test data fall into different linear regions. By comparing angle and Euclidean distance between the linear coefficients of test region and the training region activated in the extracted function that is only composed of linear regions from training data, we witnessed a significant difference. We finally measure their difference also in terms of the similarity of activation patterns to further verify that linear regions of test data are not similar to linear regions of training data, so that generalization performance of neural networks cannot be explained by a similar activation pattern either. (Note that for a region with fixed activation pattern, the network function is linear and defines the corresponding linear region and its linear coefficients.) Table 5 shows the overlap of activated neurons per layer for a CNN of type *Conv*. For each test data point, we record the activation pattern of the network when passing through this test sample, and compare it to the activation pattern of the network for the training point into which linear region the test sample falls after extraction. That is, for each ReLU and Maxpooling (MP) layer, we determine the fraction of neurons that are both active in train and test region or both

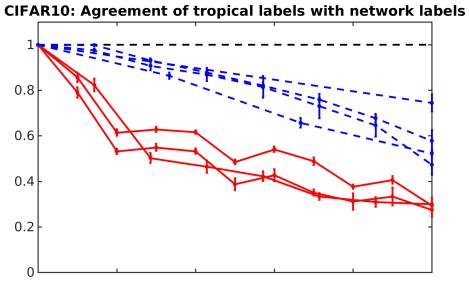



Figure 10: Average agreement with the labels computed by the neural network for all CIFAR10 networks while being transformed into a tropical function. Fully-connected networks as dotted blue (dark) lines, convolutional networks as full red (light) lines. The curves show a significant difference between the network types.

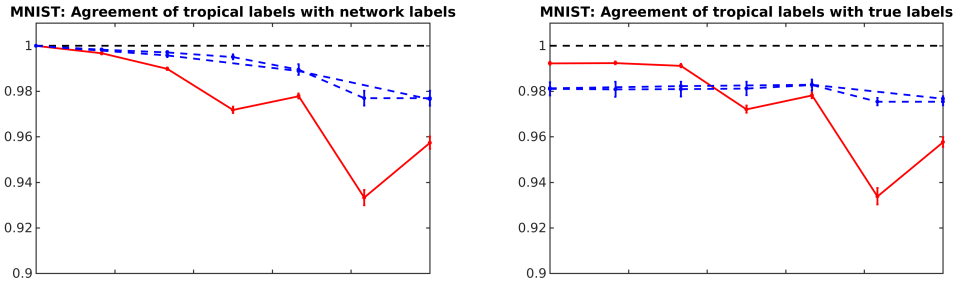



Figure 11: Average agreement with the labels computed by the neural network (left) and average test accuracy (right) for all MNIST networks while being transformed into a tropical function. Fully-connected networks as dotted blue (dark) lines, convolutional networks as full red (light) lines. The curves show a significant difference between the network types.

inactive (hence the difference of the value to 1.0 is the fraction where one neuron is active and the other one inactive.) We record the average, minimum and maximum over all training points, as well as compute the average only over those test points that are assigned the same label after extraction and the average over the test points that are assigned a different label. While the maximum shows the possibility that a test sample induces a similar activation pattern as a training sample, there are significant differences on average.

### E.6 Comparing Network Coefficients after Network Retraining

Figure 12 shows the similarity between linear coefficients belonging to training data for two separately trained networks at different stages in training. We train two networks of identical architecture and apply TropEx to extract the linear regions of all training data. For each training point, we extract the vector of linear coefficients of its linear region and measure the similarity between the coefficient vectors of the two networks in terms of (i) angle and (ii) Euclidean distance. The figure shows the distribution of points in the angle-distance space over all training data. The CNN architecture is chosen as *Conv* and the FCN architecture is chosesn as *FCN8*. While the coefficient vectors for the CNN before and after re-training are close to orthogonal and relatively far in distance, the angles for the FCN are clearly shifted to smaller angles and smaller distances, illustrating that linear region coefficients of two separately trained FCNs are related.

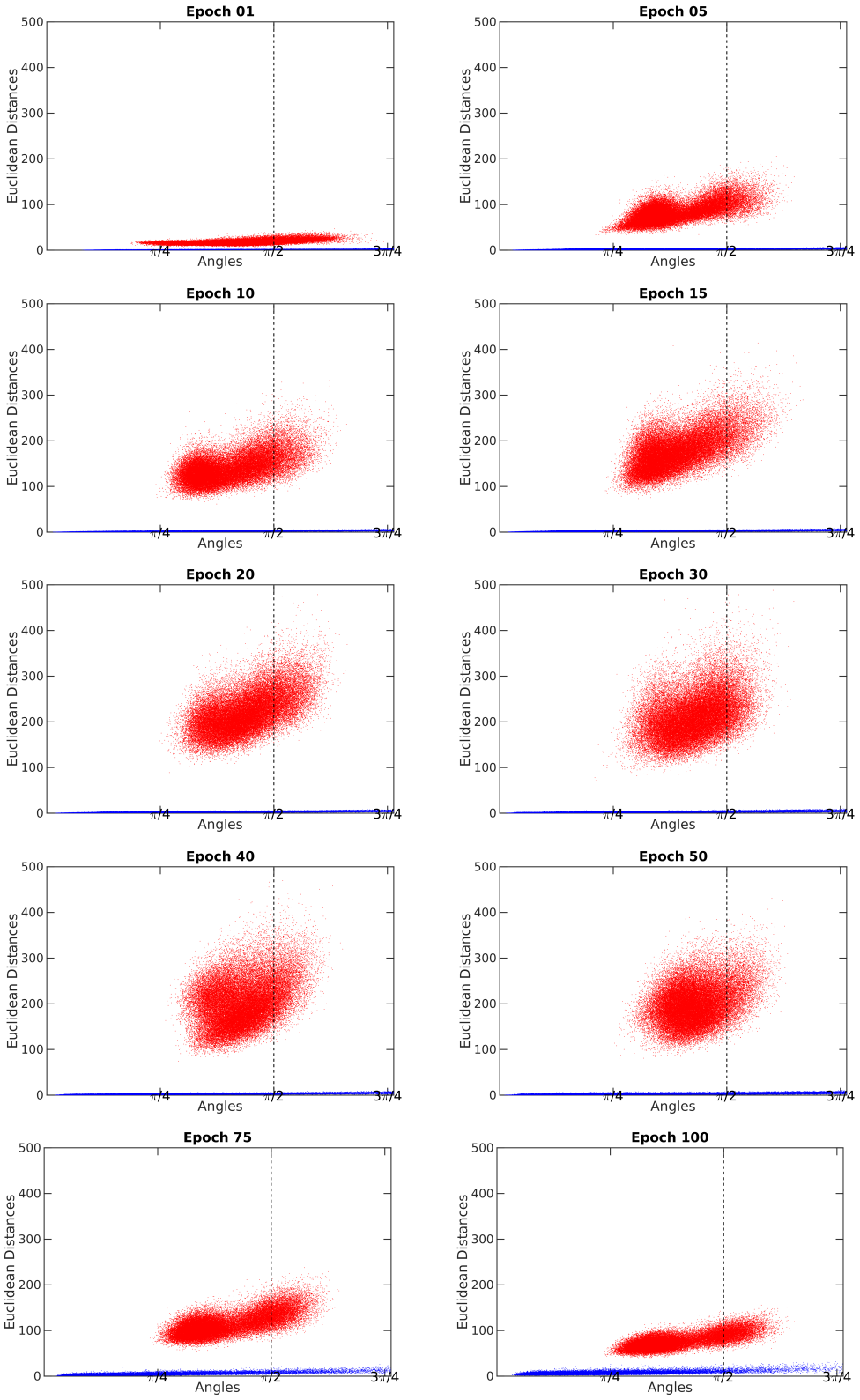



Figure 12: Angle and distance between coefficients of the linear regions of training data before and after retraining the neural network. All angles for CNNs (blue) are close to orthogonal, while linear region coefficients of FCNs (red) are clearly related.

| | ReLU | ReLU | MP | ReLU | ReLU | MP | ReLU | ReLU | MP | ReLU |
|---|---|---|---|---|---|---|---|---|---|---|
| min | 0.57 | 0.56 | 0.74 | 0.52 | 0.65 | 0.74 | 0.55 | 0.77 | 0.78 | 0.46 |
| avg | 0.79 | 0.76 | 0.81 | 0.68 | 0.76 | 0.81 | 0.69 | 0.83 | 0.84 | 0.72 |
| max | 0.97 | 0.98 | 0.96 | 0.96 | 0.97 | 0.97 | 0.97 | 0.98 | 0.97 | 0.99 |
| same | 0.81 | 0.78 | 0.82 | 0.71 | 0.79 | 0.82 | 0.73 | 0.84 | 0.85 | 0.78 |
| diff | 0.79 | 0.75 | 0.81 | 0.66 | 0.75 | 0.8 | 0.67 | 0.82 | 0.83 | 0.69 |

Table 5: Fraction of overlap in activation patterns of test linear region with region containing training data that test samples fall into after extraction. avg: average over all datapoints; min/max: minimal and maximal fraction of overlap per layer; same: average overlap over all samples that are classified with the same label as the network after extraction; diff: average overlap of all samples that are classified by a different label as the network after extraction.

### E.7 Visualization of Coefficients

We visualize the coefficients of the linear regions for the convolutional network *Conv* and the fully-connected network *FCN8* for labeled training samples. Figure 13 shows the mean value of the linear coefficients over color channels for training samples from CIFAR10 of the classes 'Airplane', 'Cat', 'Horse' and 'Ship'. We notice significant differences between the convolutional and the fully-connected network, and the fully-connected network partially allows to detect the label on close inspection, although this visual signal seems small.

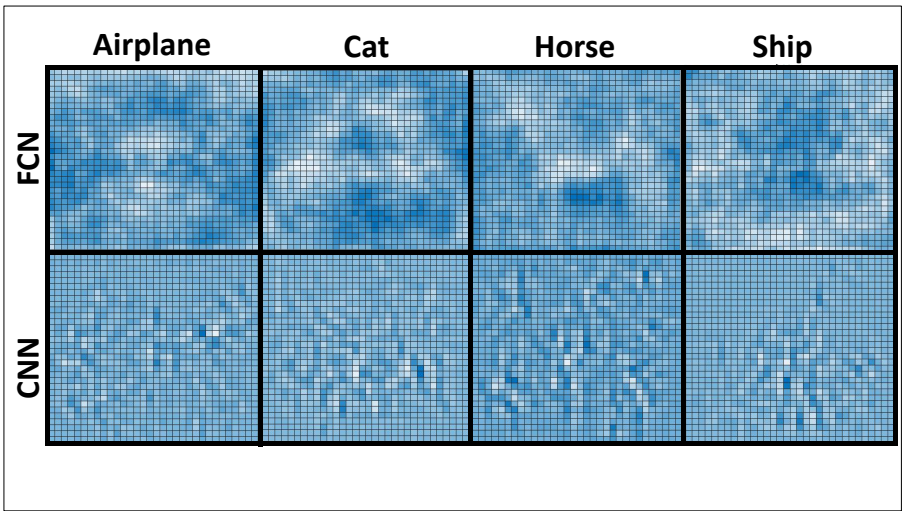



Figure 13: Heatmap of linear coefficients of training samples for the CNN *Conv* and the fully-connected network *FCN8* for four different classes of CIFAR10. Mean over color channels. Blue color represents positive and white color negative coefficients.

### E.8 Exploring the neighborhood and estimating volume and number of linear regions in practice

Since test samples never fall into regions of training samples (Section 4; Counting of Linear Regions) and since function coefficients of testing regions are significantly different to training samples, in particular for convolutional networks (Section 4; Examining Function Coefficients), we explore the following questions:

(i) How smooth is label assignment? - Despite the well-known existence of adversarial examples, label assignment may be relatively smooth. What is the label assignment of neighboring regions?
(ii) What is the volume of linear regions?
(iii) How many linear regions are there in practice? - The bounds for the maximal number of linear

regions suggests an astronomically high number of linear regions. Furthermore, the fully-connected network *Wide* has the same number of neurons in each layer as the convolutional network *Narrow* and *NarrowStrides*, hence the maximal number of linear regions is higher for the fully-connected network than for the convolutional network in this case. Is also the number of linear regions observed in practice higher for the fully-connected network?

We explore these questions by conducting the following experiment to study the neighborhood of training regions. We sampled training images $\mathbf{x}^{tr}$ of the CIFAR10 dataset and picked a random direction $d$ of length 1. We chose steps s in a range between $10^{-8}$ and 30 with exponential increments and computed the activation patterns and the network label of $\mathbf{x}^{tr} + s \cdot d$. We recorded the smallest distance at which the activation pattern changed and when the label changed. Note that it is impossible to miss earlier changes of acitvation patterns as it is impossible that, moving along a line with direction $d$, the activation pattern changes and then changes back to a previous activation pattern. This is different to label assignment since label assignment may possibly switch within a linear region, but with samples placed densely enough we expect not to miss any label changes.

Table 6 shows for different architecture types the minimal, maximal, (arithmetic) mean and geometric mean for the distance of witnessing new linear regions and label changes. For label changes, we observed a few 'adversarial examples' at small distances. For example, the distance of $0.0034$ for the network *Conv* corresponds to changing each pixel by less than 1 in the range $[0, 255]$. But overall, the label assignment is smooth with an average distance of $8$, corresponding to changing each pixel by $37$ in the range $[0, 255]$. In $11.6\%$ of cases the label never changed (*Conv*). We noticed that the network assigned the same label to almost all points on the boundary of the image hypercube $[0, 1]^{32x32x3}$, explaining why some label assignments never change.

The linear regions already changed at much smaller distances, For example for *NarrowStride*, observing new linear regions at mean distances of $10^{-3}$ implies that linear regions change before any pixel can change by 1 in the range of [0,255], casting more doubt on how meaningful the number of linear regions can be for generalization performance,

Measuring distances to neighboring linear regions also lets us estimate the volume of the linear regions and therefore estimate the number of linear regions in the image domain. The mean distance to the boundary of a linear region being $\alpha$, we estimate the volume of each linear region by $(2\alpha)^{(3 \cdot 32^2)}$. Using the average volume, we can fill the image domain $[0, 1]^{32 \times 32 \times 3}$ by linear regions of that volume and count how many are necessary ($= \alpha^{-3072}$), which we also record in Table 6. Of course, this can only give a rough estimate, for example since linear regions of training samples may differ in size to other linear regions and since the estimated volume of the regions based on mean distances is only a sample-based approximate result. We record only the order of magnitude that results from our calculations, which indeed suggests astronomically high numbers.

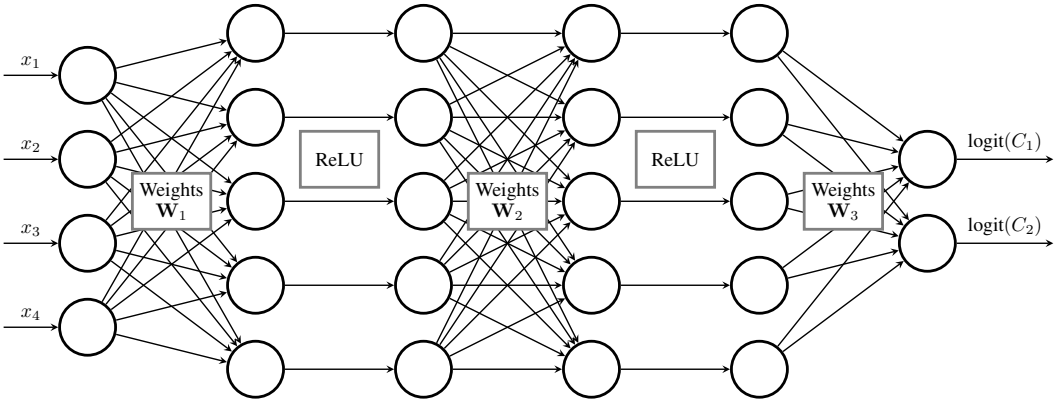
| | **Distance to neighboring regions** | | | | **Distance to new label assignment** | | | **Number linear linear regions** |
|---|---|---|---|---|---|---|---|---|
| **Network** | (arithm) mean | geom. mean | min | max | mean | min | constant label (%) | (Estimate) |
| **Conv** | 2.7e-4 | 8.4e-5 | 2.2e-8 | 2.9e-3 | 7.7 | 3.4e-3 | 12.6 | $10^{10,000}$ |
| **Narrow** | 7.2e-4 | 2.9e-4 | 2.3e-6 | 6.6e-3 | 6.9 | 4.6e-2 | 9.8 | $10^{8730}$ |
| **NarrowStride** | 3.2e-3 | 1.9e-3 | 3.0e-7 | 2.1e-2 | 11.1 | 5.7e-2 | 12.4 | $10^{6745}$ |
| **Wide** | 7.5e-3 | 3.6 e-3 | 1.2e-7 | 6.9e-2 | 18.9 | 5.7e-2 | 58.8 | $10^{5610}$ |
| **FCN8** | 2.0e-2 | 1.0e-2 | 9.7e-7 | 1.2e-1 | 19.1 | 9.0e-1 | 51.4 | $10^{4285}$ |

Table 6: Distance to neighboring linear regions and to new label assignment from training samples along a random direction. Numbers show the minimum, maximum and mean over 500 samples for different architectures. Mean distance is used to calculate a rough estimate of the number of regions.

We observe that the number of linear regions are larger for the convolutional networks *Conv* and *NarrowStride* than for the fully-connected networks *Wide* and *FCN8*. This is remarkable since *NarrowStride* and *Wide* have the same number of hidden neurons in each layer and hence the theoretically obtainable maximal number of linear regions is higher for *Wide* than for *NarrowStride*.

Also, we note that the estimated number of linear regions is not always a good measure for network performance on test data as *FCN8* performs better than *Wide* on CIFAR10 according to Table 4, but our estimate predicts a larger number of linear regions for *Wide*.

## F EXAMPLE: APPLYING TROPEX TO A TOY NETWORK




We present an example of how to apply TropEx to a toy neural network as shown in the figure. The network has three linear dense layers, with the first two being followed by a ReLU activation. During training, the last layer is followed by a softmax activation. Since the softmax function does not change the order of the outputs, we can leave it away without changing the classification results. The outputs are then the logits of the two classes $C_1$ and $C_2$. Suppose that the weight matrices are

$$\mathbf{W}_1 = \begin{pmatrix} 1 & -1 & 0 & -1 \\ -1 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ -1 & 1 & -1 & 0 \\ -1 & 0 & 1 & 1 \end{pmatrix}; \mathbf{W}_2 = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & -1 & 1 \\ 1 & -1 & 0 & 1 & 1 \\ 0 & -1 & 0 & 1 & 0 \\ -1 & -1 & 0 & 0 & 0 \end{pmatrix}; \mathbf{W}_3 = \begin{pmatrix} 1 & 0 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & 1 \end{pmatrix};$$

and the bias vectors are given by

$$\mathbf{b}_1 = \begin{pmatrix} -1 \\ 0 \\ 1 \\ -1 \\ 0 \end{pmatrix}; \quad \mathbf{b}_2 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}; \quad \mathbf{b}_3 = \begin{pmatrix} -1 \\ 0 \end{pmatrix}.$$

Suppose we are given a data set

$$\mathcal{X} = \{\mathbf{X}_1, \mathbf{X}_2\}$$

containing matrices

$$\mathbf{X}_1 = \begin{pmatrix} -1 & 0 & 0 & 1 \\ -1 & 1 & -1 & 1 \\ 0 & 1 & -1 & 1 \end{pmatrix}; \quad \mathbf{X}_2 = \begin{pmatrix} 1 & 1 & 1 & -1 \\ -1 & 0 & 0 & -1 \\ -1 & -1 & -1 & -1 \\ -1 & 1 & -1 & -1 \end{pmatrix}$$

that have 4-dimensional data points of labels 1 and 2, respectively, as their rows. Consequently,

$$\mathbf{W}_3^{\text{neg}} = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix}; \quad \mathbf{b}_3^{\text{neg}} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

Summing along the columns gives

$$\mathbf{C}^- = (0 \quad 1 \quad 2 \quad 1 \quad 0); \quad \mathbf{c}^- = (1).$$

Adding these matrices to $\mathbf{W}_3$ and $\mathbf{b}_3$ gives

$$\mathbf{C}^+ = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 2 & 1 \end{pmatrix}; \quad \mathbf{c}^+ = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Repeating the $\mathbf{C}$-matrices along the rows gives

$$\mathbf{A}_1^+ = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \end{pmatrix}; \quad \mathbf{a}_1^+ = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix};$$

$$\mathbf{A}_2^+ = \begin{pmatrix} 1 & 0 & 1 & 2 & 1 \\ 1 & 0 & 1 & 2 & 1 \\ 1 & 0 & 1 & 2 & 1 \\ 1 & 0 & 1 & 2 & 1 \end{pmatrix}; \quad \mathbf{a}_2^+ = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix};$$

$$\mathbf{A}^- = \begin{pmatrix} 0 & 1 & 2 & 1 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 1 & 2 & 1 & 0 \end{pmatrix}; \quad \mathbf{a}^- = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}.$$

Vertically stacking $\mathbf{A}_1^+, \mathbf{A}_2^+$ and $\mathbf{A}^-$ and taking the maximum along their columns gives the row vector

$$\mathbf{A}_{\text{max}} = (1 \quad 1 \quad 2 \quad 2 \quad 1)\,.$$

Let $\ell$ now denote the ReLU activation after the second dense layer. Denoting by $\overline{\ell}$ all of the network up to and including $\ell$ gives us

$$\overline{\ell}(\mathbf{X}_1) = \begin{pmatrix} 2 & 4 & 0 & 0 & 0 \\ 6 & 3 & 0 & 0 & 0 \\ 5 & 3 & 0 & 0 & 0 \end{pmatrix}; \quad \overline{\ell}(\mathbf{X}_2) = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 4 & 0 & 1 & 2 & 0 \end{pmatrix}.$$

We use the 0-elements of these matrices to set the corresponding elements of the $\mathbf{A}$-matrices to 0. The result is

$$\mathbf{A}_1^+ = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}; \quad \mathbf{a}_1^+ = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix};$$

$$\mathbf{A}_2^+ = \begin{pmatrix} 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 1 & 0 & 0 & 2 & 0 \\ 1 & 0 & 1 & 2 & 0 \end{pmatrix}; \quad \mathbf{a}_2^+ = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix};$$

$$\mathbf{A}^- = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 2 & 1 & 0 \end{pmatrix}; \quad \mathbf{a}^- = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}.$$

Hence

$$K = \mathbf{A}_{\text{max}}\mathbf{W}_2^{\text{neg}} = (1 \quad 5 \quad 0 \quad 1 \quad 0)\,; \quad \mathbf{A}_{\text{max}} = K + \mathbf{A}_{\text{max}}\mathbf{W}_2^{\text{pos}} = (4 \quad 7 \quad 2 \quad 6 \quad 3)\,.$$

Updating the $\mathbf{A}$-matrices according to $\mathbf{a} = \mathbf{a} + \mathbf{A}\mathbf{b}_2; \mathbf{A} = K + \mathbf{A}\mathbf{W}_2$ gives us

$$\mathbf{A}_1^+ = \begin{pmatrix} 2 & 7 & 2 & 1 & 1 \\ 2 & 7 & 2 & 1 & 1 \\ 2 & 7 & 2 & 1 & 1 \end{pmatrix}; \quad \mathbf{a}_1^+ = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix};$$

$$\mathbf{A}_2^+ = \begin{pmatrix} 1 & 4 & 1 & 4 & 0 \\ 1 & 3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 4 & 0 \\ 2 & 3 & 1 & 5 & 1 \end{pmatrix}; \quad \mathbf{a}_2^+ = \begin{pmatrix} 3 \\ 3 \\ 3 \\ 3 \end{pmatrix};$$

$$\mathbf{A}^- = \begin{pmatrix} 2 & 6 & 1 & 0 & 1 \\ 2 & 6 & 1 & 0 & 1 \\ 2 & 6 & 1 & 0 & 1 \\ 2 & 5 & 1 & 1 & 1 \\ 1 & 4 & 0 & 2 & 0 \\ 2 & 5 & 1 & 1 & 1 \\ 3 & 2 & 0 & 4 & 2 \end{pmatrix}; \quad \mathbf{a}^- = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 2 \\ 2 \\ 2 \end{pmatrix}.$$

Let $\ell$ now denote the ReLU activation after the first dense layer. Denoting by $\overline{\ell}$ all of the network up to and including $\ell$ gives us

$$\overline{\ell}(\mathbf{X}_1) = \begin{pmatrix} 0 & 2 & 0 & 0 & 2 \\ 0 & 3 & 1 & 2 & 1 \\ 0 & 2 & 2 & 1 & 0 \end{pmatrix}; \quad \overline{\ell}(\mathbf{X}_2) = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 2 & 0 \end{pmatrix}.$$

We use the 0-elements of these matrices to set the corresponding elements of the $\mathbf{A}$-matrices to 0. The result is

$$\mathbf{A}_1^+ = \begin{pmatrix} 0 & 7 & 0 & 0 & 1 \\ 0 & 7 & 2 & 1 & 1 \\ 0 & 7 & 2 & 1 & 0 \end{pmatrix}; \quad \mathbf{a}_1^+ = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix};$$

$$\mathbf{A}_2^+ = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 3 & 1 & 5 & 0 \end{pmatrix}; \quad \mathbf{a}_2^+ = \begin{pmatrix} 3 \\ 3 \\ 3 \\ 3 \end{pmatrix};$$

$$\mathbf{A}^- = \begin{pmatrix} 0 & 6 & 0 & 0 & 1 \\ 0 & 6 & 1 & 0 & 1 \\ 0 & 6 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 4 & 0 \end{pmatrix}; \quad \mathbf{a}^- = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 2 \\ 2 \\ 2 \end{pmatrix}.$$

Hence

$$K = \mathbf{A}_{\text{max}} \mathbf{W}_1^{\text{neg}} = (1 \quad 5 \quad 0 \quad 1 \quad 0); \quad \mathbf{A}_{\text{max}} = K + \mathbf{A}_{\text{max}} \mathbf{W}_1^{\text{pos}} = (4 \quad 7 \quad 2 \quad 6 \quad 3).$$

Updating the $\mathbf{A}$-matrices according to $\mathbf{a} = \mathbf{a} + \mathbf{A}\mathbf{b}_1; \mathbf{A} = K + \mathbf{A}\mathbf{W}_1$ gives us

$$\mathbf{A}_1^+ = \begin{pmatrix} 8 & 11 & 9 & 12 \\ 9 & 12 & 6 & 12 \\ 10 & 12 & 5 & 11 \end{pmatrix}; \quad \mathbf{a}_1^+ = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix};$$

$$\mathbf{A}_2^+ = \begin{pmatrix} 17 & 4 & 7 & 4 \\ 16 & 4 & 8 & 4 \\ 17 & 4 & 7 & 4 \\ 9 & 12 & 2 & 7 \end{pmatrix}; \quad \mathbf{a}_2^+ = \begin{pmatrix} 4 \\ 3 \\ 4 \\ -1 \end{pmatrix};$$

$$\mathbf{A}^- = \begin{pmatrix} 9 & 10 & 9 & 11 \\ 10 & 10 & 8 & 11 \\ 11 & 10 & 7 & 10 \\ 17 & 4 & 7 & 4 \\ 16 & 4 & 8 & 4 \\ 17 & 4 & 7 & 4 \\ 10 & 10 & 4 & 6 \end{pmatrix}; \quad \mathbf{a}^- = \begin{pmatrix} 1 \\ 2 \\ 2 \\ 3 \\ 2 \\ 3 \\ -2 \end{pmatrix},$$

which is the output of the algorithm.

## G POTENTIAL FOR FUTURE WORK

• *Adversarial Attacks.* Since large linear regions add adversarial robustness (Croce et al., 2018) and TropEx outputs a network function with maximally enlarged linear regions, it may provide a tool for studying various adversarial attacks. It would also be interesting to see whether adversarial attacks can harm the performance of the extracted function $\mathcal{N}^{(\mathcal{X})}$ on training data.
• *Network Pruning.* Our experiments show that removing the structure from a network $\mathcal{N}$ causes a significant drop in test accuracy while keeping the training accuracy unchanged. Zhang et al. (2018) show that tropical rational maps are the same as ReLU networks, so it is possible to obtain a new architecture $\mathcal{N}'$ from $\mathcal{N}^{(\mathcal{X})}$. From a perspective of linear regions, TropEx outputs a minimal function that is unchanged on all training data. Since different network architectures can describe the same network function, it remains to understand what additional structure needs to be imposed on the smaller architecture to obtain high test accuracy as well. This opens the possibility for a pruning technique with unchanged predictions on the data it was trained on.
• *Generalization theory.* Deep neural networks can learn data perfectly (Zhang et al., 2017), which results in a network function that correctly classifies any training sample. For such over-parameterized models, Belkin et al. (2018) study the performance on test data from a viewpoint of interpolation. Our extracted function $\mathcal{N}^{(\mathcal{X})}$ agrees with the network $\mathcal{N}$ on the data set $\mathcal{X}$, hence their difference in generalization must depend on how they interpolate between the training data $\mathcal{X}$. By observing consistent differences between CNNs and FCNs, our results suggest that the architecture plays a large role in this interpolation process. Tailored experiments with TropEx can give more insight on the connection between network architecture and interpolation performance and support the derivation of bounds on the generalization error for neural networks.