Mathematical Programming (2020) 183:3–39
https://doi.org/10.1007/s10107-020-01474-5

FULL LENGTH PAPER

Series B

Strong mixed-integer programming formulations for trained neural networks

Ross Anderson[1] · Joey Huchette[2] · Will Ma[3] · Christian Tjandraatmadja[1] · Juan Pablo Vielma[1,4]

Received: 1 June 2019 / Accepted: 23 January 2020 / Published online: 13 February 2020

Abstract

We present strong mixed-integer programming (MIP) formulations for high-dimensional piecewise linear functions that correspond to trained neural networks. These formulations can be used for a number of important tasks, such as verifying that an image classification network is robust to adversarial inputs, or solving decision problems where the objective function is a machine learning model. We present a generic framework, which may be of independent interest, that provides a way to construct sharp or ideal formulations for the maximum of d affine functions over arbitrary polyhedral input domains. We apply this result to derive MIP formulations for a number of the most popular nonlinear operations (e.g. ReLU and max pooling) that are strictly stronger than other approaches from the literature. We corroborate this computationally, showing that our formulations are able to offer substantial improvements in solve time on verification tasks for image classification networks.

An extended abstract version of this paper appeared in [4].

✉ Joey Huchette
joehuchette@rice.edu

Ross Anderson
rander@google.com

Will Ma
wm2428@gsb.columbia.edu

Christian Tjandraatmadja
ctjandra@google.com

Juan Pablo Vielma
jvielma@mit.edu; jvielma@google.com

[1] Google Inc, Cambridge, MA, USA

[2] Rice University, Houston, TX, USA

[3] Columbia University, New York, NY, USA

[4] Massachusetts Institute of Technology, Cambridge, MA, USA

Keywords Mixed-integer programming · Formulations · Deep learning

Mathematics Subject Classification 90C11

1 Introduction

Deep learning has proven immensely powerful at solving a number of important predictive tasks arising in areas such as image classification, speech recognition, machine translation, and robotics and control [32,48]. The workhorse model in deep learning is the feedforward network $\texttt{NN} : \mathbb{R}^{m_0} \to \mathbb{R}^{m_s}$ that maps a (possibly very high-dimensional) input $x^0 \in \mathbb{R}^{m_0}$ to an output $x^s = \texttt{NN}(x^0) \in \mathbb{R}^{m_s}$. A feedforward network with s layers can be recursively described as

$$x^i_j = \texttt{NL}^{i,j}(w^{i,j} \cdot x^{i-1} + b^{i,j}) \quad \forall i \in [\![s]\!],\ j \in [\![m_i]\!] \tag{1}$$

where $[\![n]\!] \stackrel{\text{def}}{=} \{1, \ldots, n\}$, m_i is both the number of *neurons* in layer i and the output dimension of the neurons in layer $i-1$ (with the input $x^0 \in \mathbb{R}^{m_0}$ considered to be the 0-th layer). Furthermore, for each $i \in [\![s]\!]$ and $j \in [\![m_i]\!]$, $\texttt{NL}^{i,j} : \mathbb{R} \to \mathbb{R}$ is some simple nonlinear *activation function* that is fixed before training, and $w^{i,j}$ and $b^{i,j}$ are the *weights* and *bias* of an affine function which is learned during the training procedure. In its simplest and most common form, the activation function would be the rectified linear unit (ReLU), defined as $\texttt{ReLU}(v) = \max\{0, v\}$.

Many standard nonlinearities NL, such as the ReLU, are *piecewise linear*: that is, there exists a partition $\{S^i \subseteq D\}_{i=1}^d$ of the domain and affine functions $\{f^i\}_{i=1}^d$ such that $\texttt{NL}(x) = f^i(x)$ for all $x \in S^i$. If all nonlinearities describing NN are piecewise linear, then the entire network NN is piecewise linear as well, and conversely, any continuous piecewise linear function can be represented by a ReLU neural network [5]. Beyond the ReLU, we also investigate activation functions that can be expressed as the maximum of finitely many affine functions, a class which includes certain common operations such as max pooling and reduce max.

There are numerous contexts in which one may want to solve an optimization problem containing a trained neural network such as NN. For example, this paradigm arises in deep reinforcement learning problems with high dimensional action spaces and where any of the cost-to-go function, immediate cost, or the state transition functions are learned by a neural network [6,23,56,62,64,80]. More generally, this approach may be used to approximately optimize learned functions that are difficult to model explicitly [34,52,53]. Alternatively, there has been significant recent interest in verifying the robustness of trained neural networks deployed in systems like self-driving cars that are incredibly sensitive to unexpected behavior from the machine learning model [20,60,68]. Relatedly, a string of recent work has used optimization over neural networks trained for visual perception tasks to *generate* new images which are "most representative" for a given class [59], are "dreamlike" [57], or adhere to a particular artistic style via neural style transfer [30].

1.1 MIP formulation preliminaries

In this work, we study mixed-integer programming (MIP) approaches for optimization problems containing trained neural networks. In contrast to heuristic or local search methods often deployed for the applications mentioned above, MIP offers a framework for producing provably optimal solutions. This is particularly important, for example, in verification problems, where rigorous dual bounds can guarantee robustness in a way that purely primal methods cannot.

Let $f : D \subseteq \mathbb{R}^\eta \to \mathbb{R}$ be a η-variate affine function with domain D, $\texttt{NL} : \mathbb{R} \to \mathbb{R}$ be an univariate nonlinear activation function, and $\circ$ be the standard function composition operator so that $(\texttt{NL} \circ f)(x) = \texttt{NL}(f(x))$. Functions of the form $(\texttt{NL} \circ f)(x)$ precisely describe an individual neuron, with η inputs and one output. A standard way to model a function $g : D \subseteq \mathbb{R}^\eta \to \mathbb{R}$ using MIP is to construct a formulation of its graph defined by $\text{gr}(g; D) \stackrel{\text{def}}{=} \{(x, y) \in \mathbb{R}^\eta \times \mathbb{R} \,|\, x \in D,\ y = g(x)\}$. We therefore focus on constructing MIP formulations for the *graph* of individual neurons:

$$\text{gr}(\texttt{NL} \circ f; D) \stackrel{\text{def}}{=} \left\{(x, y) \in \mathbb{R}^\eta \times \mathbb{R} \,|\, x \in D,\ y = (\texttt{NL} \circ f)(x)\right\}, \tag{2}$$

because, as we detail at the end of this section, we can readily produce a MIP formulation for the entire network as the composition of formulations for each individual neuron.

Definition 1 Throughout, we will notationally use the convention that $x \in \mathbb{R}^\eta$ and $y \in \mathbb{R}$ are respectively the *input* and *output* variables of the function $g : D \subseteq \mathbb{R}^\eta \to \mathbb{R}$ we are modeling. In addition, $v \in \mathbb{R}^h$ and $z \in \mathbb{R}^q$ are respectively any auxiliary continuous and binary variables used to construct a formulation of $\text{gr}(g; D)$. The orthogonal projection operator Proj will be subscripted by the variables to project onto, e.g. $\text{Proj}_{x,y}(R) = \{(x, y) \,|\, \exists v, z \text{ s.t. } (x, y, v, z) \in R\}$ is the orthogonal projection of R onto the "original space" of x and y variables. We denote by $\text{ext}(Q)$ the set of extreme points of a polyhedron Q.

Take $S \stackrel{\text{def}}{=} \text{gr}(g; D) \subseteq \mathbb{R}^{\eta+1}$ to be the set we want to model, and a polyhedron $Q \subset \mathbb{R}^{\eta+1+h+q}$. Then:

- A *(valid) mixed-integer programming (MIP) formulation* of S consists of the linear constraints on $(x, y, v, z) \in \mathbb{R}^{\eta+1+h+q}$ which define a polyhedron Q, combined with the integrality constraints $z \in \{0, 1\}^q$, such that

$$S = \text{Proj}_{x,y}\left(Q \cap (\mathbb{R}^{\eta+1+h} \times \{0, 1\}^q)\right).$$

  We refer to Q as the *linear programming (LP) relaxation* of the formulation.
- A MIP formulation is *sharp* if $\text{Proj}_{x,y}(Q) = \text{Conv}(S)$.
- A MIP formulation is *hereditarily sharp* if fixing any subset of binary variables z to 0 or 1 preserves sharpness.
- A MIP formulation is *ideal* (or *perfect*) if $\text{ext}(Q) \subseteq \mathbb{R}^{\eta+1+h} \times \{0, 1\}^q$.
- The *separation* problem for a family of inequalities is to find a valid inequality violated by a given point or certify that no such inequality exists.

– An inequality is *valid* for the formulation if each integer feasible point in $\underline{Q} = \{(x, y, v, z) \in Q \mid z \in \{0, 1\}^q\}$ satisfies the inequality. Moreover, a valid inequality is *facet-defining* if the dimension of the points in $\underline{Q}$ satisfying the inequality at equality is exactly one less than the dimension of $\underline{Q}$ itself.

Note that ideal formulations are sharp, but the converse is not necessarily the case [72, Proposition 2.4]. In this sense, ideal formulations offer the tightest possible relaxation, and the integrality property in Definition 1 tends to lead to superior computational performance. Furthermore, note that a hereditarily sharp formulation is a formulation which retains its sharpness at every node in a branch-and-bound tree, and as such is potentially superior to a formulation which only guarantees sharpness at the root node [41,42]. Additionally, it is important to keep in mind that modern MIP solvers will typically require an explicit, finite list of inequalities defining Q.

Finally, for each $i \in [\![s]\!]$ and $j \in [\![m_i]\!]$ let $Q_{i,j} \subseteq \mathbb{R}^{m_{i-1}+1+h_{i,j}+q_{i,j}}$ be a polyhedron such that $\text{gr}(\texttt{NL}^{i,j} \circ f^{i,j}; D_{i,j}) = \text{Proj}_{x,y}\left(Q_{i,j} \cap (\mathbb{R}^{m_{i-1}+1+h_{i,j}} \times \{0, 1\}^{q_{i,j}})\right)$, where each $D_{i,j}$ is the domain of neuron $\texttt{NL}^{i,j}$ and $f^{i,j}(x) = w^{i,j} \cdot x + b^{i,j}$ is the learned affine function in network (1). Then, a formulation for the complete network (1) is given by

$$\left(x^{i-1}, x^i_j, v^{i,j}, z^{i,j}\right) \in \left(Q_{i,j} \cap (\mathbb{R}^{m_{i-1}+1+h_{i,j}} \times \{0, 1\}^{q_{i,j}})\right) \quad \forall i \in [\![s]\!],\ j \in [\![m_i]\!].$$

Any properties of the individual-neuron formulations $Q_{i,j}$ (e.g. being ideal), are not necessarily preserved for the combined formulation for the complete network. However, for similarly sized-formulations, combining stronger formulations (e.g. ideal instead of sharp) usually leads to complete formulations that are more computationally effective [72]. Hence, our focus for all theoretical properties of the formulations will be restricted to individual-neuron formulations.

1.2 Our contributions

We highlight the contributions of this work as follows.

1. *Generic recipes for building strong MIP formulations of the maximum of d affine functions for any bounded polyhedral input domain.*
   – [Propositions 3 and 4] We derive both primal and dual characterizations for ideal formulations via the *Cayley embedding*.
   – [Propositions 5 and 6] We relax the Cayley embedding in a particular way, and use this to derive simpler primal and dual characterizations for (hereditarily) sharp formulations.
   – We discuss how to separate both dual characterizations via subgradient descent.
2. *Simplifications for common special cases.*
   – [Corollary 1] We show the equivalence of the ideal and sharp characterizations when $d = 2$ (i.e. the maximum of two affine functions).

- [Proposition 7] We show that, if the input domain is a product of simplices, the separation problem of the sharp formulation can be reframed as a series of transportation problems.
- [Corollaries 2 and 3, and Proposition 9] When the input domain is a product of simplices, and either (1) $d = 2$, or (2) each simplex is two-dimensional, we provide an explicit, finite description for the sharp formulation. Furthermore, none of these inequalities are redundant.

3. *Application of these results to construct MIP formulations for neural network nonlinearities.*
   - [Propositions 12 and 13] We derive an explicit ideal formulation for the ReLU nonlinearity over a box input domain, the most common case. Separation over this ideal formulation can be performed in time linear in the input dimension.
   - [Corollary 5] We derive an explicit ideal formulation for the ReLU nonlinearity where some (or all) of the input domain is *one-hot encoded* categorical or discrete data. Again, the separation can be performed efficiently, and none of the inequalities are redundant.
   - [Propositions 10 and 11] We present an explicit hereditarily sharp formulation for the maximum of d affine functions over a box input domain, and provide an efficient separation routine. Moreover, a subset of the defining constraints serve as a tightened big-M formulation.
   - [Proposition 14] We produce similar results for more exotic neurons: the leaky ReLU and the clipped ReLU (see the online supplement).
4. *Computational experiments on verification problems arising from image classification networks trained on the MNIST digit data set.*
   - We observe that our new formulations, along with just a few rounds of separation over our families of cutting planes, can improve the solve time of Gurobi on verification problems by orders of magnitude.

Our contributions are depicted in Fig. 1. It serves as a roadmap of the paper.

1.3 Relevant prior work

In recent years a number of authors have used MIP formulations to model trained neural networks [19,22,24,29,40,47,54,64,66,67,70,80,81], mostly applying big-M formulation techniques to ReLU-based networks. When applied to a single neuron of the form (2), these big-M formulations will not be ideal or offer an exact convex relaxation; see Example 1 for an illustration. Additionally, a stream of literature in the deep learning community has studied convex relaxations [12,26,27,61,63], primarily for verification tasks. Moreover, some authors have investigated how to use convex relaxations within the training procedure in the hopes of producing neural networks with a priori robustness guarantees [25,78,79].

The usage of mathematical programming in deep learning, specifically for training predictive models, has also been investigated in [15]. Beyond mathematical programming and convex relaxations, a number of authors have investigated other algorithmic

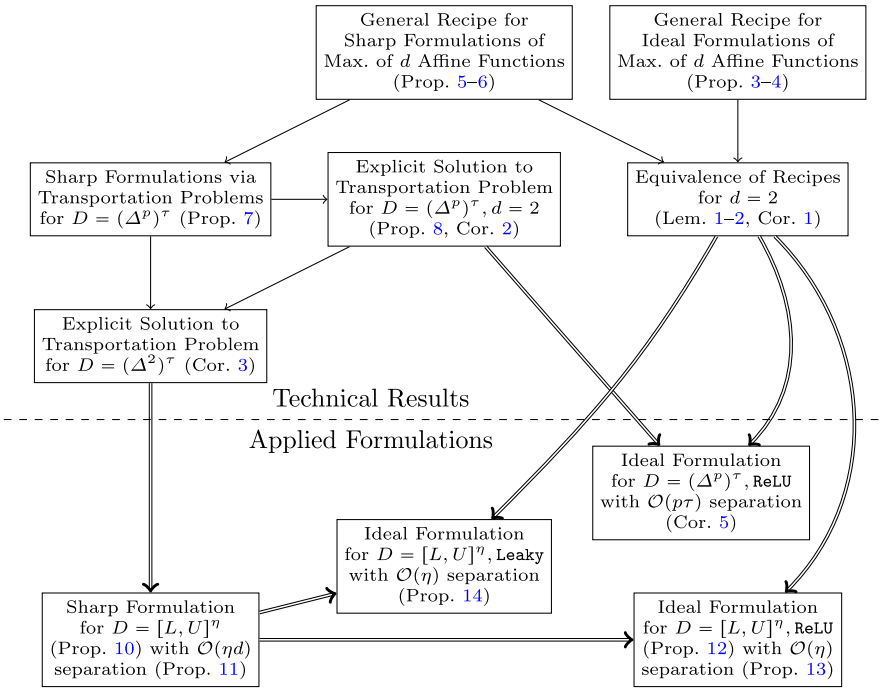


Fig. 1 A roadmap of our results. The single arrows depict dependencies between our technical results, while the double arrows depict the way in which we use these results to establish our applied formulations in the paper. Notationally, d is the number of pieces of the piecewise linear function, $[L, U]^\eta$ is an η-dimensional box, and $(\Delta^p)^\tau$ refers to a product of τ simplices, each with p components

techniques for modeling trained neural networks in optimization problems, drawing primarily from the satisfiability, constraint programming, and global optimization communities [10,11,43,51,65]. Throughout this stream of the literature, there is discussion of the performance for specific subclasses of neural network models, including binarized [44] or input convex [3] neural networks. Improving our formulations for specific subclasses of neural networks would constitute interesting future work.

Outside of applications in machine learning, the formulations presented in this paper also have connections with existing structures studied in the MIP and constraint programming communities, like indicator variables, on/off constraints, and convex envelopes [7,13,17,37,38,69]. Finally, this paper has connections with distributionally robust optimization, in that the primal–dual pair of sharp formulations presented can be viewed as equivalent to the discrete sup–sup duality result [21] for the marginal distribution model [35,58,77].

1.4 Starting assumptions and notation

We use the following notational conventions throughout the paper.

– The nonnegative orthant: $\mathbb{R}_{\geqslant 0} \stackrel{\text{def}}{=} \{x \in \mathbb{R} \mid x \geqslant 0\}$.

- The n-dimensional simplex: $\Delta^n \stackrel{\text{def}}{=} \left\{x \in \mathbb{R}^n_{\geqslant 0} \mid \sum_{i=1}^n x_i = 1\right\}$.
- The set of integers from 1 to n: $[\![n]\!] = \{1, \ldots, n\}$.
- "Big-M" coefficients: $M^+(f; D) \stackrel{\text{def}}{=} \max_{x \in D} f(x)$ and $M^-(f; D) \stackrel{\text{def}}{=} \min_{x \in D} f(x)$.
- The dilation of a set: if $z \in \mathbb{R}_{\geqslant 0}$ and $D \subseteq \mathbb{R}^\eta$, then $z \cdot D \stackrel{\text{def}}{=} \{zx \mid x \in D\}$.[1]

Furthermore, we will make the following simplifying assumptions.

Assumption 1 The input domain D is a bounded polyhedron.

While a bounded input domain assumption will make the formulations and analysis considerably more difficult than the unbounded setting (see [7] for a similar phenomenon), it ensures that standard MIP representability conditions are satisfied (e.g. [72, Sect. 11]). Furthermore, variable bounds are natural for many applications (for example in verification problems), and are absolutely essential for ensuring reasonable dual bounds.

Assumption 2 Each neuron is *irreducible*: for any $k \in [\![d]\!]$, there exists some $x \in D$ where $f^k(x) > f^\ell(x)$ for each $\ell \neq k$.

Observe that if a neuron is not irreducible, this means that it is unnecessarily complex, and one or more of the affine functions can be completely removed. Moreover, the assumption can be verified in polynomial time via d LPs by checking if $\max_{x,\Delta} \left\{\Delta \,\middle|\, x \in D,\ \Delta \leqslant f^k(x) - f^\ell(x)\ \forall \ell \neq k\right\} > 0$ for each $k \in [\![d]\!]$. In the special case where $d = 2$ (e.g. ReLU) and D is a box, this can be done in linear time. Finally, if the assumption does not hold, it will not affect the validity of the formulations or cuts derived in this work, though certain results pertaining to non-redundancy or facet-defining properties may no longer hold.

2 Motivating example: the ReLU nonlinearity over a box domain

Since the work of Glorot et al. [31], the ReLU neuron has become the workhorse of deep learning models. Despite the simplistic ReLU structure, neural networks formed by ReLU neurons are easy to train, reason about, and can model any continuous piecewise linear function [5]. Moreover, they can approximate any continuous, nonlinear function to an arbitrary precision under a bounded width [36].

Accordingly, the general problem of maximizing (or minimizing) the output of a trained ReLU network is NP-hard [43]. Nonetheless, in this section we present the strongest possible MIP formulations for a single ReLU neuron without continuous auxiliary variables. As discussed at the end of Sect. 1.1 and corroborated in the computational study in Sect. 6, this leads to faster solve times for the entire ReLU network in practice.

2.1 A big-*M* formulation

To start, we will consider the ReLU in the simplest possible setting: where the input is univariate. Take the two-dimensional set $\mathrm{gr}(\texttt{ReLU}; [l, u])$, where $[l, u]$ is some interval

[1] Note that if $D = \left\{x \in \mathbb{R}^\eta \,\middle|\, Ax \leqslant b\right\}$ is polyhedral, then $z \cdot D = \left\{x \in \mathbb{R}^\eta \,\middle|\, Ax \leqslant bz\right\}$.

in $\mathbb{R}$ containing zero. It is straightforward to construct an ideal formulation for this univariate ReLU.

Proposition 1 *An ideal formulation for* $\mathrm{gr}(\texttt{ReLU}; [l, u])$ *is:*

$$y \geqslant x \tag{3a}$$
$$y \leqslant x - l(1 - z) \tag{3b}$$
$$y \leqslant uz \tag{3c}$$
$$(x, y, z) \in \mathbb{R} \times \mathbb{R}_{\geqslant 0} \times [0, 1] \tag{3d}$$
$$z \in \{0, 1\}. \tag{3e}$$

Proof Follows from inspection, or as a special case of Proposition 12 to be presented in Sect. 5.2. □

A more realistic setting would have an η-variate ReLU nonlinearity whose input is some η-variate affine function $f : [L, U] \to \mathbb{R}$ where $L, U \in \mathbb{R}^\eta$ and $[L, U] \stackrel{\text{def}}{=} \left\{x \in \mathbb{R}^\eta \,\middle|\, L_i \leqslant x_i \leqslant U_i \ \forall i \in [\![\eta]\!]\right\}$ (i.e. the η-variate ReLU nonlinearity given by $\texttt{ReLU} \circ f$). The *box-input* $[L, U]$ corresponds to known (finite) bounds on each component, which can typically be efficiently computed via interval arithmetic or other standard methods.

Observe that we can model the graph of the *η-variate ReLU neuron* as a simple composition of the graph of a univariate ReLU activation function and an η-variate affine function:

$$\mathrm{gr}\left(\texttt{ReLU} \circ f; [L, U]\right) = \left\{(x, y) \in \mathbb{R}^{\eta+1} \,\middle|\, \begin{array}{c} (f(x), y) \in \mathrm{gr}\left(\texttt{ReLU}; [m^-, m^+]\right) \\ L \leqslant x \leqslant U \end{array} \right\}, \tag{4}$$

where $m^- \stackrel{\text{def}}{=} M^-(f; [L, U])$ and $m^+ \stackrel{\text{def}}{=} M^+(f; [L, U])$. Using formulation (3) as a submodel, we can write a formulation for the ReLU over a box domain as:

$$y \geqslant f(x) \tag{5a}$$
$$y \leqslant f(x) - M^-(f; [L, U]) \cdot (1 - z) \tag{5b}$$
$$y \leqslant M^+(f; [L, U]) \cdot z \tag{5c}$$
$$(x, y, z) \in [L, U] \times \mathbb{R}_{\geqslant 0} \times [0, 1] \tag{5d}$$
$$z \in \{0, 1\}. \tag{5e}$$

This is the approach taken recently in the bevy of papers referenced in Sect. 1.3. Unfortunately, after the composition with the affine function f over a box input domain, this formulation is no longer sharp.

Example 1 If $f(x) = x_1 + x_2 - 1.5$, formulation (5) for $\mathrm{gr}(\texttt{ReLU} \circ f; [0, 1]^2)$ is

$$y \geqslant x_1 + x_2 - 1.5 \tag{6a}$$

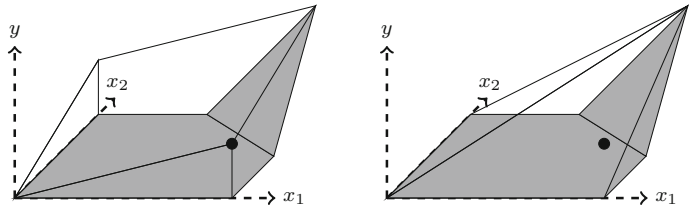


Fig. 2 For $f(x) = x_1 + x_2 - 1.5$: (Left) Conv(gr($\texttt{ReLU} \circ f$; $[0, 1]^2$)), and (Right) the projection of the big-M formulation (5) to (x, y)-space, where we mark the point $(x, y) = ((1, 0), 0.25)$ that is not in the convex hull, but is valid for the projection of the big-M LP relaxation (6a)–(6d)

$$y \leqslant x_1 + x_2 - 1.5 + 1.5(1 - z) \tag{6b}$$
$$y \leqslant 0.5z \tag{6c}$$
$$(x, y, z) \in [0, 1]^2 \times \mathbb{R}_{\geqslant 0} \times [0, 1] \tag{6d}$$
$$z \in \{0, 1\}. \tag{6e}$$

The point $(\hat{x}, \hat{y}, \hat{z}) = ((1, 0), 0.25, 0.5)$ is feasible for the LP relaxation (6a)–(6d). However, observe that the inequality $y \leqslant 0.5x_2$ is valid for gr($\texttt{ReLU} \circ f$; $[0, 1]^2$), but is violated by $(\hat{x}, \hat{y})$. Therefore, the formulation does not offer an exact convex relaxation (and, hence, is not ideal). See Fig. 2 for an illustration: on the left, of the big-M formulation projected to (x, y)-space, and on the right, the tightest possible convex relaxation.

Moreover, the integrality gap of (5) can be arbitrarily bad, even in fixed dimension η.

Example 2 Fix $\gamma \in \mathbb{R}_{\geqslant 0}$ and even $\eta \in \mathbb{N}$. Take the affine function $f(x) = \sum_{i=1}^{\eta} x_i$, the input domain $[L, U] = [-\gamma, \gamma]^\eta$, and $\hat{x} = \gamma \cdot (1, -1, \ldots, 1, -1)$ as a scaled vector of alternating positive and negative ones. We can check that $(\hat{x}, \hat{y}, \hat{z}) = (\hat{x}, \frac{1}{2}\gamma\eta, \frac{1}{2})$ is feasible for the LP relaxation of the big-M formulation (5). Additionally, $f(\hat{x}) = 0$, and for any $\tilde{y}$ such that $(\hat{x}, \tilde{y}) \in \text{Conv}(\text{gr}(\texttt{ReLU} \circ f; [L, U]))$, then $\tilde{y} = 0$ necessarily. Therefore, there exists a fixed point $\hat{x}$ in the input domain where the tightest possible convex relaxation (for example, from a sharp formulation) is exact, but the big-M formulation deviates from this value by at least $\frac{1}{2}\gamma\eta$.

Intuitively, this example suggests that the big-M formulation can be particularly weak around the boundary of the input domain, as it cares only about the value $f(x)$ of the affine function, and not the particular input value x.

2.2 An ideal extended formulation

It is possible to produce an ideal *extended* formulation for the ReLU neuron by introducing a modest number of auxiliary continuous variables:

$$(x, y) = (x^0, y^0) + (x^1, y^1) \tag{7a}$$
$$y^0 = 0 \geqslant w \cdot x^0 + b(1 - z) \tag{7b}$$

$$y^1 = w \cdot x^1 + bz \geqslant 0 \tag{7c}$$
$$L(1-z) \leqslant x^0 \leqslant U(1-z) \tag{7d}$$
$$Lz \leqslant x^1 \leqslant Uz \tag{7e}$$
$$z \in \{0, 1\}, \tag{7f}$$

This is the standard "multiple choice" formulation for piecewise linear functions [75], which can also be derived from techniques due to Balas [8,9].

Although the multiple choice formulation offers the tightest possible convex relaxation for a single neuron, it requires a copy x^0 of the input variables [the copy x^1 can be eliminated using the Eq. (7a)]. This means that when this formulation is applied to every neuron in the network to formulate NN, the total number of continuous variables required is $m_0 + \sum_{i=1}^{r}(m_{i-1}+1)m_i$, where m_i is the number of neurons in layer i. In contrast, the big-M formulation requires only $m_0 + \sum_{i=1}^{r} m_i$ continuous variables to formulate the entire network. As we will see in Sect. 6, the quadratic growth in size of the extended formulation can quickly become burdensome. Additionally, a folklore observation in the MIP community is that multiple choice formulations tend to not perform as well as expected in simplex-based branch-and-bound algorithms, likely due to degeneracy introduced by the block structure of the formulation [74].

2.3 An ideal MIP formulation without auxiliary continuous variables

In this work, our most broadly useful contribution is the derivation of an ideal MIP formulation for the ReLU nonlinearity over a box domain that is non-extended; that is, it does not require additional auxiliary variables as in formulation (7). We informally summarize this main result as follows.

Main result for ReLU networks (informal)

There exists an explicit ideal nonextended formulation for the η-variate ReLU nonlinearity over a box domain, i.e. it requires only a single auxiliary binary variable. It has an exponential (in η) number of inequality constraints, each of which are facet-defining. However, it is possible to separate over this family of inequalities in time scaling linearly in η.

We defer the formal statement and proof to Sect. 5.2, for after we have derived the requisite machinery. This is also the main result for the extended abstract version of this work [4], where it is derived through alternative means.

3 Our general machinery: formulations for the maximum of *d* affine functions

We will state our main structural results in the following generic setting. Take the maximum operator $\texttt{Max}(v_1, \ldots, v_d) = \max_{i=1}^{d} v_i$ over d scalar inputs. We study the

composition of this nonlinearity with d affine functions $f^i : D \to \mathbb{R}$ with $f^i(x) = w^i \cdot x + b^i$, all sharing some bounded polyhedral domain D:

$$S_{\max} \stackrel{\text{def}}{=} \text{gr}(\texttt{Max} \circ (f^1, \ldots, f^d); D) \equiv \left\{ (x, y) \in D \times \mathbb{R} \,\middle|\, y = \max_{i=1}^{d} f^i(x) \right\},$$

This setting subsumes the ReLU over box input domain presented in Sect. 2 as a special case with $d = 2$, $f^2(x) = 0$, and $D = [L, U]$. It also covers a number of other settings arising in modern deep learning, either by making D more complex (e.g. one-hot encodings for categorical features), or by increasing d (e.g. max pooling neurons often used in convolutional networks for image classification tasks [18], or in maxout networks [33]).

In this section, we present structural results that characterize the *Cayley embedding* [39,73,74,76] of $S_{\max}$. Take the set

$$S_{\text{cayley}} \stackrel{\text{def}}{=} \bigcup_{k=1}^{d} \left\{ (x, f^k(x), \mathbf{e}^k) \,|\, x \in D_{|k} \right\},$$

where $\mathbf{e}^k$ is the unit vector where the k-th element is 1, and for each $k \in [\![d]\!]$,

$$D_{|k} \stackrel{\text{def}}{=} \left\{ x \in D \,\middle|\, k \in \arg\max_{\ell=1}^{d} f^\ell(x) \right\} \equiv \left\{ x \in D \,\middle|\, w^k \cdot x + b^k \geqslant w^\ell \cdot x + b^\ell \; \forall \ell \neq k \right\}$$

is the portion of the input domain D where f^k attains the maximum. The *Cayley embedding* of $S_{\max}$ is the convex hull of this set: $R_{\text{cayley}} \stackrel{\text{def}}{=} \text{Conv}(S_{\text{cayley}})$.

The following straightforward observation holds directly from definition.

Observation 1 *The set R_{cayley} is a bounded polyhedron, and an ideal formulation for $S_{\max}$ is given by the system $\left\{ (x, y, z) \in R_{\text{cayley}} \,\middle|\, z \in \{0, 1\}^d \right\}$.*

Therefore, if we can produce an explicit inequality description for R_{cayley}, we immediately have an ideal MIP formulation for $S_{\max}$. Indeed, we have already seen an extended representation in (7) when $d = 2$, which we now state in the more general setting (projecting out the superfluous copies of y).

Proposition 2 *An ideal MIP formulation for $S_{\max}$ is:*

$$(x, y) = \sum_{k=1}^{d} (\tilde{x}^k, w^k \cdot \tilde{x}^k + b^k z_k) \tag{8a}$$

$$\tilde{x}^k \in z_k \cdot D_{|k} \qquad \forall k \in [\![d]\!] \tag{8b}$$

$$z \in \Delta^d \tag{8c}$$

$$z \in \{0, 1\}^d. \tag{8d}$$

Denote its LP relaxation by $R_{\text{extended}} = \left\{ (x, y, z, \tilde{x}^1, \ldots, \tilde{x}^k) \,\middle|\, (8a - 8c) \right\}$. Then $\text{Proj}_{x,y,z}(R_{\text{extended}}) = R_{\text{cayley}}$.

Proof Follows directly from results in [8,9]. □

Although this formulation is ideal and polynomially-sized, this extended formulation can exhibit poor practical performance, as noted in Sect. 2.2 and corroborated in the computational experiments in Sect. 6. Observe that, from definition, constraint (8b) for a given $k \in [\![d]\!]$ is equivalent to the set of constraints $\widetilde{x}^k \in z_k \cdot D$ and $w^k \cdot \widetilde{x}^k + b^k z_k \geqslant w^\ell \cdot \widetilde{x}^k + b^\ell z_k$ for each $\ell \neq k$.

3.1 A recipe for constructing ideal formulations

Our goal in this section is to derive generic tools that allow us to build ideal formulations for $S_{\max}$ via the Cayley embedding.

3.1.1 A primal characterization

Our first structural result provides a characterization for the Cayley embedding. Although it is not an explicit polyhedral characterization, we will subsequently see how it can be massaged into a more practically amenable form.

Take the system

$$y \leqslant \overline{g}(x, z) \tag{9a}$$
$$y \geqslant \underline{g}(x, z) \tag{9b}$$
$$(x, y, z) \in D \times \mathbb{R} \times \Delta^d, \tag{9c}$$

where

$$\overline{g}(x, z) \stackrel{\text{def}}{=} \max_{\widetilde{x}^1, \ldots, \widetilde{x}^d} \left\{ \sum_{k=1}^{d} w^k \cdot \widetilde{x}^k + b^k z_k \;\middle|\; \begin{array}{l} x = \sum_k \widetilde{x}^k \\ \widetilde{x}^k \in z_k \cdot D_{|k} \quad \forall k \in [\![d]\!] \end{array} \right\}$$
$$\underline{g}(x, z) \stackrel{\text{def}}{=} \min_{\widetilde{x}^1, \ldots, \widetilde{x}^d} \left\{ \sum_{k=1}^{d} w^k \cdot \widetilde{x}^k + b^k z_k \;\middle|\; \begin{array}{l} x = \sum_k \widetilde{x}^k \\ \widetilde{x}^k \in z_k \cdot D_{|k} \quad \forall k \in [\![d]\!] \end{array} \right\},$$

and define the set $R_{\text{ideal}} \stackrel{\text{def}}{=} \{(x, y, z) \,|\, (9)\}$. Note that R_{ideal} can be thought of as (8) with the $\widetilde{x}^k$ variables implicitly projected out.

Proposition 3 *The set R_{ideal} is polyhedral, and $R_{\text{ideal}} = R_{\text{cayley}}$.*

Proof By Proposition 2, it suffices to show that $R_{\text{ideal}} = \text{Proj}_{x,y,z}(R_{\text{extended}})$. We start by observing that, as $\overline{g}$ (respectively $\underline{g}$) is concave (resp. convex) in its imputs as it is the value function of a linear program (cf. [14, Theorem 5.1]). Therefore, the set of points satisfying (9) is convex.

Let $(\hat{x}, \hat{y}, \hat{z})$ be an extreme point of R_{ideal}, which exists as R_{ideal} is convex. Then it must satisfy either (9a) or (9b) at equality, as otherwise it is a convex combination of $(\hat{x}, \hat{y} - \epsilon, \hat{z})$ and $(\hat{x}, \hat{y} + \epsilon, \hat{z})$ for some $\epsilon > 0$. Take $\widetilde{x}^1, \ldots, \widetilde{x}^d$ that optimizes $\overline{g}(x, z)$ or $\underline{g}(x, z)$, depending on which constraint is satisfied at equality. Then

$(\hat{x}, \hat{y}, \hat{z}, \widetilde{x}^1, \ldots, \widetilde{x}^d) \in R_{\text{extended}}$. In other words, $\text{ext}(R_{\text{ideal}}) \subseteq \text{Proj}_{x,y,z}(R_{\text{extended}})$, and thus $R_{\text{ideal}} \subseteq \text{Proj}_{x,y,z}(R_{\text{extended}})$ by convexity.

Conversely, let $(\hat{x}, \hat{y}, \hat{z}, \widetilde{x}^1, \ldots, \widetilde{x}^d) \in R_{\text{extended}}$. Then $\hat{y} = \sum_{k=1}^{d}(w^k \cdot \widetilde{x}^k + b^k \hat{z}_k) \leqslant \overline{g}(\hat{x}, \hat{z})$, as $(\{\widetilde{x}^k\}_{k=1}^{d})$ is feasible for the optimization problem in $\overline{g}(x, z)$. Likewise, $\hat{y} \geqslant \underline{g}(\hat{x}, \hat{z})$, and (9c) is trivially satisfied. Therefore, $(\hat{x}, \hat{y}, \hat{z}) \in R_{\text{ideal}}$.

Polyhedrality of R_{ideal} then follows as R_{cayley} is itself a polyhedron. □

Note that the input domain constraint $x \in D$ is implied by the constraints (9a)–(9b), and therefore do not need to be explicitly included in this description. However, we include it here in our description for clarity. Moreover, observe that $\overline{g}$ is a function from $D \times \Delta^d$ to $\mathbb{R} \cup \{-\infty\}$, since the optimization problem may be infeasible, but is always bounded from above since D is bounded. Likewise, $\underline{g}$ is a function from $D \times \Delta^d$ to $\mathbb{R} \cup \{+\infty\}$.

3.1.2 A dual characterization

From Proposition 3, we can derive a more useful characterization by applying Lagrangian duality to the LPs describing the envelopes $\overline{g}(x, z)$ and $\underline{g}(x, z)$.

Proposition 4 *The Cayley embedding R_{cayley} is equal to all (x, y, z) satisfying*

$$y \leqslant \overline{\alpha} \cdot x + \sum_{k=1}^{d} \left(\max_{x^k \in D_{|k}} \{(w^k - \overline{\alpha}) \cdot x^k\} + b^k \right) z_k \quad \forall \overline{\alpha} \in \mathbb{R}^\eta \tag{10a}$$

$$y \geqslant \underline{\alpha} \cdot x + \sum_{k=1}^{d} \left(\min_{x^k \in D_{|k}} \{(w^k - \underline{\alpha}) \cdot x^k\} + b^k \right) z_k \quad \forall \underline{\alpha} \in \mathbb{R}^\eta \tag{10b}$$

$$(x, y, z) \in D \times \mathbb{R} \times \Delta^d. \tag{10c}$$

Proof Consider the upper bound inequalities for y in Proposition 3, that is, (9a). Now apply the change of variables $x^k \leftarrow \frac{\widetilde{x}^k}{z_k}$ for each $k \in [\![d]\!]$ and, for all (x, z), take the Lagrangian dual of the optimization problem in $\overline{g}(x, z)$ with respect to the constraint $x = \sum_k x^k z_k$. Note that the duality gap is zero since the problem is an LP. We then obtain that

$$\begin{aligned}
\overline{g}(x, z) &= \min_{\overline{\alpha}} \max_{x^k \in D_{|k}} \sum_{k=1}^{d} \left(w^k \cdot x^k + b^k \right) z_k + \overline{\alpha} \cdot \left(x - \sum_{k=1}^{d} x^k z_k \right) \\
&= \min_{\overline{\alpha}} \ \overline{\alpha} \cdot x + \sum_{k=1}^{d} \left(\max_{x^k \in D_{|k}} \{(w^k - \overline{\alpha}) \cdot x^k\} + b^k \right) z_k.
\end{aligned}$$

In other words, we can equivalently express (9a) via the family of inequalities (10a). The same can be done with (9b), yielding the set of inequalities (10b). Therefore, $\left\{(x, y, z) \,\middle|\, (10)\right\} = \text{Proj}_{x,y,z}(R_{\text{extended}}) = R_{\text{cayley}}$. □

This gives an exact outer description for the Cayley embedding in terms of an infinite number of linear inequalities. Despite this, the formulation enjoys a simple, interpretable form: we can view the inequalities as choosing coefficients on x and individually tightening the coefficients on z according to explicitly described LPs. Similar to the relationship between (8) and (9), (10) can be seen as a simplification of the standard cut generation LP for unions of polyhedra [8,9]. In later sections, we will see that this decoupling is helpful to simplify (a variant of) this formulation for special cases.

Separating a point $(\hat{x}, \hat{y}, \hat{z})$ over (10a) can be done by evaluating $\overline{g}(\hat{x}, \hat{z})$ in the form (10a) (and in the analogous form of $\underline{g}(\hat{x}, \hat{z})$ for (10b)). As typically done when using Lagrangian relaxation, this optimization problem can be solved via a subgradient or bundle method, where each subgradient can be computed by solving the inner LP in (10a) for all $x^k \in D_{|k}$. Observe that any feasible solution $\overline{\alpha}$ for the optimization problem in (10a) yields a valid inequality. However, this optimization problem is unbounded when $(\hat{x}, \hat{z}) \notin \text{Proj}_{x,z}(R_{\text{cayley}})$ (i.e. when the primal form of $\overline{g}(\hat{x}, \hat{z})$ is infeasible). In other words, as illustrated in Fig. 3, $\overline{g}$ is an *extended* real valued function such that $\overline{g}(x, z) \in \mathbb{R} \cup \{-\infty\}$, so care must be taken to avoid numerical instabilities when separating a point $(\hat{x}, \hat{z})$ where $\overline{g}(\hat{x}, \hat{z}) = -\infty$.[2]

3.2 A recipe for constructing hereditarily sharp formulations

Although Proposition 4 gives a separation-based way to optimize over $S_{\max}$, there are two potential downsides to this approach. First, it does not give us a explicit, finite description for a MIP formulation that we can directly pass to a MIP solver. Second, the separation problem requires optimizing over $D_{|k}$, which may be substantially more complicated than optimizing over D (for example, if D is a box).

Therefore, in this section we set our sights slightly lower and present a similar technique to derive *sharp* MIP formulations for $S_{\max}$. Furthermore, we will see that our formulations trivially satisfy the hereditary sharpness property. In the coming sections, we will see how we can deploy these results in a practical manner, and study settings in which the simpler sharp formulation will also, in fact, be ideal.

3.2.1 A primal characterization

Consider the system

$$y \leqslant \overline{h}(x, z) \tag{11a}$$

$$y \geqslant w^k \cdot x + b^k \quad \forall k \in [\![d]\!] \tag{11b}$$

$$(x, y, z) \in D \times \mathbb{R} \times \Delta^d, \tag{11c}$$

[2] As is standard in a Benders' decomposition approach, we can address this by adding a *feasibility cut* describing the domain of $\overline{g}$ (the region where it is finite valued) instead of an *optimality cut* of the form (10a).

where

$$\overline{h}(x, z) \stackrel{\text{def}}{=} \max_{\widetilde{x}^1,\ldots,\widetilde{x}^d} \left\{ \sum_{k=1}^{d} (w^k \cdot \widetilde{x}^k + b^k z_k) \;\middle|\; \begin{array}{l} x = \sum_k \widetilde{x}^k \\ \widetilde{x}^k \in z_k \cdot D \quad \forall k \in [\![d]\!] \end{array} \right\}.$$

Take the set $R_{\text{sharp}} \stackrel{\text{def}}{=} \big\{(x, y, z) \;\big|\; (11)\big\}$.

It is worth dwelling on the differences between the systems (9) and (11). First, we have completely replaced the constraint (9b) with d explicit linear inequalities (11b). Second, when replacing $\overline{g}$ with $\overline{h}$ we have replaced the inner maximization over $D_{|k}$ with an inner maximization over D (modulo constant scaling factors). As we will see in Sect. 5.1, this is particularly advantageous when D is trivial to optimize over (for example, a simplex or a box), allowing us to write these constraints in closed form, whereas optimizing over $D_{|k}$ may be substantially more difficult (i.e. requiring an LP solve).

Furthermore, we will show that while (11) is not ideal, it is hereditarily sharp, and so in general may offer a strictly stronger relaxation than a standard sharp formulation. In particular, the formulation may be stronger than a sharp formulation constructed by composing a big-M formulation, along with an exact convex relaxation in the (x, y)-space produced, for example, by studying the upper concave envelope of the function $\texttt{Max} \circ (f^1, \ldots, f^d)$.

Proposition 5 *The system describing* $\big\{(x, y, z) \in R_{\text{sharp}} \;\big|\; z \in \{0, 1\}^d\big\}$ *is a hereditarily sharp MIP formulation of* $S_{\max}$.

Proof For the result, we must show four properties: polyhedrality of R_{sharp}, validity of the formulation whose LP relaxation is R_{sharp}, sharpness, and then hereditary sharpness. We proceed in that order.

To show polyhedrality, consider a fixed value $(\hat{x}, \hat{y}, \hat{z})$ feasible for (11), and presume that we express the domain via the linear inequality constraints $D = \big\{x \;\big|\; Ax \leqslant c\big\}$. First, observe that due to (11a) and (11b), $\overline{h}(\hat{x}, \hat{z})$ is bounded from below. Now, using LP duality, we may rewrite

$$\begin{aligned} \overline{h}(\hat{x}, \hat{z}) &= \max_{\widetilde{x}^1,\ldots,\widetilde{x}^d} \left\{ \sum_{k=1}^{d} w^k \cdot \widetilde{x}^k \;\middle|\; \begin{array}{l} \hat{x} = \sum_k \widetilde{x}^k \\ A\widetilde{x}^k \leqslant \hat{z}_k c \quad \forall k \in [\![d]\!] \end{array} \right\} + \sum_{k=1}^{d} b^k \hat{z}_k \\ &= \min_{(\alpha, \beta^1, \ldots, \beta^k) \in R} \left\{ \alpha \cdot \hat{x} + \sum_{k=1}^{d} c \cdot \beta^k \hat{z}_k \right\} + \sum_{k=1}^{d} b^k \hat{z}_k, \end{aligned}$$

where R is a polyhedron that is *independent of* $\hat{x}$ *and* $\hat{z}$. Therefore, as (a) the above optimization problem is linear with $\hat{x}$ and $\hat{z}$ fixed, and (b) $\overline{h}(\hat{x}, \hat{z})$ is bounded from below, we may replace R with $\text{ext}(R)$ in the above optimization problem. In other words, $\overline{h}(\hat{x}, \hat{z})$ is equal to the minimum of a finite number of alternatives which are affine in $\hat{x}$ and $\hat{z}$. Therefore, $\overline{h}$ is a concave continuous piecewise linear function, and so R_{sharp} is polyhedral.

To show validity, we must have that $\text{Proj}_{x,y}\left(R_{\text{sharp}} \cap (\mathbb{R}^\eta \times \mathbb{R} \times \{0,1\}^d)\right) = S_{\max}$. Observe that if $(\hat{x}, \hat{y}, \hat{z}) \in R_{\text{sharp}} \cap (\mathbb{R}^\eta \times \mathbb{R} \times \{0,1\}^d)$, then $\hat{z} = \mathbf{e}^\ell$ for some $\ell \in [\![d]\!]$, and

$$\overline{h}(\hat{x}, \hat{z}) = \max_{\widetilde{x}^1,\ldots,\widetilde{x}^d} \left\{ \sum_{k=1}^d w^k \cdot \widetilde{x}^k + b^\ell \;\middle|\; \begin{array}{ll} \hat{x} = \sum_k \widetilde{x}^k & \\ \widetilde{x}^\ell \in D & \\ \widetilde{x}^k = \mathbf{0}^\eta & \forall k \neq \ell \end{array} \right\} = w^\ell \hat{x} + b^\ell,$$

where the first equality follows as $\widetilde{x}^k = \mathbf{0}^\eta$ for each $k \neq \ell$ (recall that if D is bounded, then $0 \cdot D = \left\{x \in \mathbb{R}^\eta \;\middle|\; Ax \leqslant 0\right\} = \{\mathbf{0}^\eta\}$). Along with (11b), this implies that $\hat{y} = w^\ell \cdot \hat{x} + b^\ell$, and that $\hat{y} \geqslant w^k \cdot \hat{x} + b^k$ for each $k \neq \ell$, giving the result.

To show sharpness, we must prove that $\text{Proj}_{x,y}(R_{\text{sharp}}) = \text{Conv}(S_{\max})$. First, recall from Proposition 2 that $\text{Conv}(S_{\max}) = \text{Proj}_{x,y}(R_{\text{extended}})$; thus, we state our proof in terms of R_{extended}. We first show that $\text{Proj}_{x,y}(R_{\text{extended}}) \subseteq \text{Proj}_{x,y}(R_{\text{sharp}})$. Take $(\hat{x}, \hat{y}, \hat{z}, \{\hat{x}^k\}_{k=1}^d) \in R_{\text{extended}}$. Then $\hat{y} = \sum_{k=1}^d (w^k \cdot \hat{x}^k + b^k \hat{z}_k) \leqslant \overline{h}(\hat{x}, \hat{z})$, as $(\{\hat{x}^k\}_{k=1}^d)$ is feasible for the optimization problem in $\overline{h}(x, z)$. It also holds that $\hat{y} \geqslant w^k \cdot \hat{x} + b^k$ for all $k \in [\![d]\!]$ and $\hat{x} \in D$ directly from the definition of $S_{\max}$, giving the result.

Next, we show that $\text{Proj}_{x,y}(R_{\text{sharp}}) \subseteq \text{Proj}_{x,y}(R_{\text{extended}})$. This proof is similar to the proof of Proposition 3, except that we choose z that simplifies the constraints. It suffices to show that $\text{ext}(\text{Proj}_{x,y}(R_{\text{sharp}})) \subseteq \text{Proj}_{x,y}(R_{\text{extended}})$. Let $(\hat{x}, \hat{y}) \in \text{ext}(\text{Proj}_{x,y}(R_{\text{sharp}}))$. Define $\overline{h}(x) \stackrel{\text{def}}{=} \max_z \left\{\overline{h}(x, z) \;\middle|\; z \in \Delta^d\right\}$. Then either $(\hat{x}, \hat{y})$ satisfies $\hat{y} = \overline{h}(\hat{x})$, or it satisfies (11b) at equality for some $k \in [\![d]\!]$, since otherwise $(\hat{x}, \hat{y})$ is a convex combination of the points $(\hat{x}, \hat{y} - \epsilon)$ and $(\hat{x}, \hat{y} + \epsilon)$ feasible for $\text{Proj}_{x,y}(R_{\text{sharp}})$ for some $\epsilon > 0$. We show that in either case, $(\hat{x}, \hat{y}) \in \text{Proj}_{x,y}(R_{\text{extended}})$.

Case 1 Suppose that for some $j \in [\![d]\!]$, $(\hat{x}, \hat{y})$ satisfies the corresponding inequality in (11b) at equality; that is, $\hat{y} = w^j \cdot \hat{x} + b^j$. Then the point $(\hat{x}, \hat{y}, \mathbf{e}_j, \{\hat{x}^k\}_{k=1}^d) \in R_{\text{extended}}$, where $\hat{x}^j = x$ and $\hat{x}^\ell = 0$ if $\ell \neq j$. Hence, $(\hat{x}, \hat{y}) \in \text{proj}_{x,y}(R_{\text{extended}})$.

Case 2 Suppose $(\hat{x}, \hat{y})$ satisfies $\hat{y} = \overline{h}(\hat{x})$. Let $\hat{z}$ be an optimal solution for the optimization problem defining $\overline{h}(\hat{x})$, and $\{\hat{x}^k\}_{k=1}^d$ be an optimal solution for $\overline{h}(\hat{x}, \hat{z})$. By design, $(\hat{x}, \hat{y}, \hat{z}, \{\hat{x}^k\}_{k=1}^d)$ satisfies all constraints in R_{extended}, except potentially constraint (8b).

We show that constraint (8b) is satisfied as well. Suppose not for contradiction; that is, $w^k \cdot \hat{x}^k + b^k \hat{z}_k < w^\ell \cdot \hat{x}^k + b^\ell \hat{z}_k$ for some pair $k, \ell \in [\![d]\!]$, $\ell \neq k$. Consider the solution $(\{\overline{x}^k\}_{k=1}^d, \overline{z})$ identical to $(\{\hat{x}^k\}_{k=1}^d, \hat{z})$ except that $\overline{z}_k = 0$, $\overline{x}^k = \mathbf{0}^\eta$, $\overline{z}_\ell = \hat{z}_\ell + \hat{z}_k$, and $\overline{x}^\ell = \hat{x}^\ell + \hat{x}^k$. By inspection, this solution is feasible for $\overline{h}(\hat{x})$. The objective value changes by $-(w^k \cdot \hat{x}^k + b^k \hat{z}_k) + (w^\ell \cdot \hat{x}^k + b^\ell \hat{z}_k) > 0$, contradicting the optimality of $(\{\hat{x}^k\}_{k=1}^d, \hat{z})$. Therefore, $(\hat{x}, \hat{y}, \hat{z}, \{\hat{x}^k\}_{k=1}^d) \in R_{\text{extended}}$, and thus $(\hat{x}, \hat{y}) \in \text{Proj}_{x,y}(R_{\text{extended}})$.

Finally, we observe that hereditary sharpness follows from the definition of $\overline{h}$. In particular, fixing any $z_k = 0$ implies that $\widetilde{x}^k = \mathbf{0}^\eta$ in the maximization problem defining $\overline{h}$. In other words, the variables $\widetilde{x}^k$ and z_k drop completely from the maximization problem defining $\overline{h}$, meaning that it is equal to the corresponding version of $\overline{h}$ with the function k completely dropped as input. Additionally, if any $z_k = 1$, then $z_\ell = 0$

for each $\ell \neq k$ since $z \in \Delta^d$, and hence $\widetilde{x}^\ell = \mathbf{0}^\eta$. In this case, $\overline{h}(x, z) = w^k \cdot x + b^k$, which gives the result. □

3.2.2 A dual characterization

We can apply a duality-based approach to produce an (albeit infinite) linear inequality description for the set R_{sharp}, analogous to Sect. 3.1.2.

Proposition 6 *The set R_{sharp} is equal to all (x, y, z) such that*

$$y \leqslant \alpha \cdot x + \sum_{k=1}^{d} \left(\max_{x^k \in D} \{(w^k - \alpha) \cdot x^k\} + b^k \right) z_k \quad \forall \alpha \in \mathbb{R}^\eta \tag{12a}$$

$$y \geqslant w^k \cdot x + b^k \quad \forall k \in [\![d]\!] \tag{12b}$$

$$(x, y, z) \in D \times \mathbb{R} \times \Delta^d. \tag{12c}$$

Proof Follows in an analogous manner as in Proposition 4. □

Figure 3 depicts slices of the functions $\overline{g}(x, z)$, $\underline{g}(x, z)$, and $\overline{h}(x, z)$, created by fixing some value of z and varying x. Observe that $\overline{g}(x, z)$ can be viewed as the largest value for y such that (x, y) can be written as a convex combination of points in the graph using convex multipliers z. Likewise, $\underline{g}(x, z)$ can be interpreted as the minimum value for y. In $\overline{h}(x, z)$, we relax $D_{|k}$ to D, and thus we can interpret it similarly to $\overline{g}(x, z)$, except that we may take convex combinations of points constructed by evaluating the affine functions at any point in the domain, not only those where the given function attains the maximum. Figure 3b shows that, in general, $\overline{h}(x, z)$ can be strictly looser than $\overline{g}(x, z)$ for $(x, z) \in \text{Proj}_{x,z}(R_{\text{cayley}})$. A similar situation occurs for the lower envelopes as illustrated by Fig. 3d. However, we prove in the next section that this does not occur for $d = 2$, along with other desirable properties in special cases.

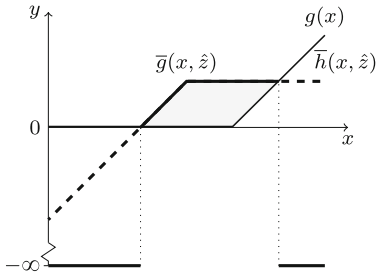
4 Simplifications to our machinery under common special cases

In this section we study how our dual characterizations in Propositions 4 and 6 simplify under common special cases with the number of input affine functions d and the input domain D.

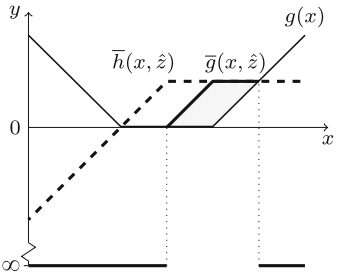
4.1 Simplifications when $d = 2$

When we consider taking the maximum of only two affine functions (i.e. $d = 2$), we can prove that R_{sharp} is, in fact, ideal.

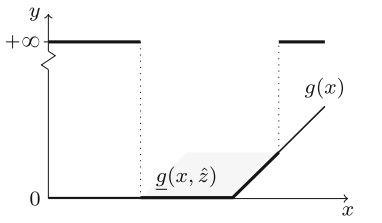
We start by returning to Proposition 3 and show that it can be greatly simplified when $d = 2$. We first show $\underline{g}(x, z)$ can be replaced by the maximum of the affine functions as illustrated in Fig. 3c, although it is not possible for $d > 2$ as seen in Fig. 3d.



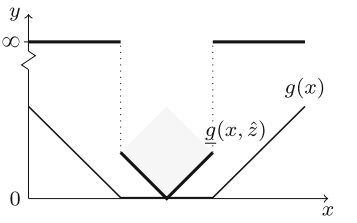
(a) Upper bounds $\overline{g}(x,\hat{z})$ and $\overline{h}(x,\hat{z})$ for $g(x) = \max_{x\in[0,2]}\{0, x-2\}$, $\hat{z} = \left(\frac{1}{2}, \frac{1}{2}\right)$.



(b) Upper bounds $\overline{g}(x,\hat{z})$ and $\overline{h}(x,\hat{z})$ for $g(x) = \max_{x\in[0,2]}\{-x+1, 0, x-2\}$, $\hat{z} = \left(0, \frac{1}{2}, \frac{1}{2}\right)$.



(c) Lower bounds $\underline{g}(x,\hat{z})$ for $g(x) = \max_{x\in[0,2]}\{0, x-2\}$, $\hat{z} = \left(\frac{1}{2}, \frac{1}{2}\right)$.



(d) Lower bounds $\underline{g}(x,\hat{z})$ for $g(x) = \max_{x\in[0,2]}\{-x+1, 0, x-2\}$, $\hat{z} = \left(\frac{1}{2}, 0, \frac{1}{2}\right)$.

Fig. 3 Examples of the functions $\overline{g}(x,z)$, $\overline{h}(x,z)$, and $\underline{g}(x,z)$ defined in (9) and (11) with some fixed value for z. Here, $g(x)$ is the "maximum" function of interest, defined below each subfigure. **a**, **c** Depict the case $d = 2$ (maximum of two affine functions), while **b**, **d** illustrate when $d = 3$, each pair emphasizing upper and lower bounds respectively. Note that $\overline{g}(x,z)$ and $\overline{h}(x,z)$ coincide in (**a**) for $x \in [1, 2.5]$, and in (**b**) for $x \in [2, 2.5]$. The thick solid lines represent $\overline{g}(x,z)$ in (**a**, **b**) and $\underline{g}(x,z)$ in (**c**, **d**), whereas the dashed lines correspond to $\overline{h}(x,z)$. The thin solid lines represent $S_{\max}$ and the shaded region is the slice of R_{cayley} with $z = \hat{z}$

Lemma 1 *If $d = 2$, then at any values of (x,z) where $\overline{g}(x,z) \geq \underline{g}(x,z)$ (i.e. there exists a y such that $(x,y,z) \in R_{\text{cayley}}$), we have*

$$\underline{g}(x,z) = \max\{w^1 \cdot x + b^1, w^2 \cdot x + b^2\}.$$

Proof For convenience, we work with the change of variables $x^k \leftarrow \frac{\widetilde{x}^k}{z_k}$ for each $k \in [\![2]\!]$. Suppose without loss of generality that $x \in D_{|2}$, and consider any feasible solution (x^1, x^2) to the optimization problem for $\underline{g}(x,z)$, which is feasible by the assumption that $(x,z) \in \text{Proj}_{x,z}(R_{\text{cayley}})$. We will show that $(w^1 \cdot x^1 + b^1)z_1 + (w^2 \cdot x^2 + b^2)z_2 \geq w^2 \cdot x + b^2$. We assume that $z_2 > 0$, since otherwise $x = x^1 \in D_{|1} \cap D_{|2}$ and the result is immediate.

Since $x = x^1 z_1 + x^2 z_2$ and $z \in \Delta^2$, the line segment joining x^1 to x^2 contains x. Furthermore, since $x^1 \in D_{|1}$ and $x^2 \in D_{|2}$, this line segment also intersects the hyperplane $\left\{\hat{x} \in \mathbb{R}^\eta \middle| w^1 \cdot \hat{x} + b^1 = w^2 \cdot \hat{x} + b^2\right\}$. Let $\hat{x}^1$ denote this point of intersection, and let $\hat{z}^1 \in \Delta^2$ be such that $\hat{x}^1 = x^1 \hat{z}^1_1 + x^2 \hat{z}^1_2$. Since $x \in D_{|2}$, we know that $\hat{x}^1$ is

closer to x^1 than x, i.e. $\hat{z}_1^1 \geq z_1$. Moreover, take the point $\hat{x}^2$ on this line segment such that $x = \hat{x}^1 z_1 + \hat{x}^2 z_2$, where $\hat{x}^2 = x^1 \hat{z}_1^2 + x^2 \hat{z}_2^2$ for some $\hat{z}^2 \in \Delta^2$. We have $\hat{z}_1^2 \leq z_1$ since $\hat{x}^2$ is further away from x^1 than x. Note that $\hat{x}^1 \in D_{|1} \cap D_{|2}$ while $\hat{x}^2 \in D_{|2}$, and thus $(\hat{x}^1, \hat{x}^2)$ is feasible.

It can be computed that $\hat{z}_1^2 = z_1 \cdot \frac{\hat{z}_2^1}{z_2}$, which implies that $z_1 = z_1(\hat{z}_1^1 + \hat{z}_2^1) = z_1 \hat{z}_1^1 + z_2 \hat{z}_1^2$ and $z_2 = z_2(\hat{z}_1^2 + \hat{z}_2^2) = z_1 \hat{z}_2^1 + z_2 \hat{z}_2^2$. Using these two identities, we obtain

$$\begin{aligned}
&(w^1 \cdot x^1 + b^1) z_1 + (w^2 \cdot x^2 + b^2) z_2 \\
&= (w^1 \cdot x^1 + b^1)(z_1 \hat{z}_1^1 + z_2 \hat{z}_1^2) + (w^2 \cdot x^2 + b^2)(z_1 \hat{z}_2^1 + z_2 \hat{z}_2^2) \\
&= ((w^1 \cdot x^1 + b^1) \hat{z}_1^1 + (w^2 \cdot x^2 + b^2) \hat{z}_2^1) z_1 \\
&\quad + ((w^1 \cdot x^1 + b^1) \hat{z}_1^2 + (w^2 \cdot x^2 + b^2) \hat{z}_2^2) z_2 \\
&= (f(x^1) \hat{z}_1^1 + f(x^2) \hat{z}_2^1) z_1 + (f(x^1) \hat{z}_1^2 + f(x^2) \hat{z}_2^2) z_2,
\end{aligned}$$

where we let $f(\hat{x})$ denote the function $\max\{w^1 \cdot \hat{x} + b^1, w^2 \cdot \hat{x} + b^2\}$, recalling that $x^1 \in D_{|1}$ and $x^2 \in D_{|2}$. Since $f(\hat{x})$ is convex, by Jensen's inequality the preceding expression is at least $f(x^1 \hat{z}_1^1 + x^2 \hat{z}_2^1) z_1 + f(x^1 \hat{z}_1^2 + x^2 \hat{z}_2^2) z_2$. The preceding expression equals $(w^2 \cdot \hat{x}^1 + b^2) z_1 + (w^2 \cdot \hat{x}^2 + b^2) z_2$ by the definitions of $\hat{x}^1$ and $\hat{x}^2$, and the fact that they both lie in $D_{|2}$. Recalling the equation $x = \hat{x}^1 z_1 + \hat{x}^2 z_2$ used to select $\hat{x}^2$ completes the proof. □

Moreover, we show that when $d = 2$ we can replace $\overline{g}$ with $\overline{h}$, as illustrated in Fig. 3a. This property may not hold when $d > 2$ as shown in Fig. 3b.

Lemma 2 *If $d = 2$, then at any values of (x, z) where $\overline{g}(x, z) \geq \underline{g}(x, z)$ (i.e. there exists a y such that $(x, y, z) \in R_{\text{cayley}}$), we have*

$$\overline{g}(x, z) = \max_{\widetilde{x}^1, \widetilde{x}^2} \left\{ w^1 \cdot \widetilde{x}^1 + b^1 z_1 + w^2 \cdot \widetilde{x}^2 + b^2 z_2 \;\middle|\; \begin{array}{l} x = \widetilde{x}^1 + \widetilde{x}^2 \\ \widetilde{x}^1 \in z_1 \cdot D \\ \widetilde{x}^2 \in z_2 \cdot D \end{array} \right\}. \tag{13}$$

Proof We show that despite expanding the feasible set of the optimization problem in (13) by replacing $D_{|k}$ by D, its optimal value is no greater when (x, z) is such that $\overline{g}(x, z) \geq \underline{g}(x, z)$. It suffices to show without loss of generality that $w^1 \cdot \widetilde{x}^1 + b^1 z_1 \geqslant w^2 \cdot \widetilde{x}^1 + b^2 z_1$ holds for any optimal $\widetilde{x}^1, \widetilde{x}^2$. By the assumption on (x, z) and Lemma 1, we have $\overline{g}(x, z) \geqslant w^2 \cdot x + b^2$, which implies the existence of some optimal $\widetilde{x}^1, \widetilde{x}^2$. That is, we have $w^2 \cdot (x - \widetilde{x}^1) + b^2 z_2 + w^1 \cdot \widetilde{x}^1 + b^1 z_1 \geqslant w^2 \cdot x + b^2$, which is equivalent to $w^1 \cdot \widetilde{x}^1 + b^1 z_1 \geqslant w^2 \cdot \widetilde{x}^1 + b^2 z_1$. □

After observing that these simplifications are identical to those presented in Proposition 6, we obtain the following corollary promised at the beginning of the section.

Corollary 1 *When $d = 2$, $\left\{(x, y, z) \in R_{\text{sharp}} \,\middle|\, z \in \{0, 1\}^d\right\}$ is an ideal MIP formulation of $S_{\max}$.*

Proof Lemmas 1 and 2 imply that $R_{\text{sharp}} = R_{\text{ideal}}$, while Proposition 3 implies that $R_{\text{ideal}} = R_{\text{cayley}}$, completing the chain and giving the result. □

In later sections, we will study conditions under which we can produce an explicit inequality description for R_{sharp}.

4.2 Simplifications when *D* is the product of simplices

In this section, we consider another important special case: when the input domain is the Cartesian product of simplices. Indeed, the box domain case introduced in Sect. 2 can be viewed as a product of two-dimensional simplices, and we will also see in Sect. 5.3 that this structure naturally arises in machine learning settings with categorical or discrete features.

When D is the product of simplices, we can derive a finite representation for the the set (12) [i.e. a finite representation for the infinite family of linear inequalities (12a)] through an elegant connection with the transportation problem. To do so, we introduce the following notation.

Definition 2 Suppose the input domain is $D = \prod_{i=1}^{\tau} \Delta^{p_i}$, with $p_1 + \cdots + p_\tau = \eta$. For notational simplicity, we re-organize the indices of x and refer to its entries via $x_{i,j}$, where $i \in [\![\tau]\!]$ is the simplex index, and $j \in [\![p_i]\!]$ refers to the coordinate within simplex i. The domain for x is then

$$D = \left\{ ((x_{i,j})_{j=1}^{p_i})_{i=1}^{\tau} \,\middle|\, (x_{i,j})_{j=1}^{p_i} \in \Delta^{p_i} \quad \forall i \in [\![\tau]\!] \right\}, \tag{14}$$

where the rows of x correspond to each simplex. Correspondingly, we re-index the weights of the affine functions so that for each $k \in [\![d]\!]$, we have $f^k(x) = \sum_{i=1}^{\tau} \sum_{j=1}^{p_i} w_{i,j}^k x_{i,j} + b^k$.

Using the notation from Definition 2, constraints (12a) can be written as

$$y \leqslant \sum_{i=1}^{\tau} \sum_{j=1}^{p_i} \alpha_{i,j} x_{i,j} + \sum_{k=1}^{d} \left(\max_{x^k \in D} \sum_{i=1}^{\tau} \sum_{j=1}^{p_i} (w_{i,j}^k - \alpha_{i,j}) x_{i,j}^k + b^k \right) z_k \quad \forall \alpha \in \mathbb{R}^\eta.$$

Since D is a product of simplices, the maximization over $x^k \in D$ appearing in the right-hand side above is separable over each simplex $i \in [\![\tau]\!]$. Moreover, for each simplex i, the maximum value of $\sum_{j=1}^{p_i} (w_{ij}^k - \alpha_{ij}) x_{ij}^k$, subject to the constraint $x^k \in D$, is obtained when $x_{ij}^k = 1$ for some $j \in [\![p_i]\!]$. Therefore, the family of constraints (12a) is equivalent to

$$\begin{aligned} y &\leqslant \min_{\alpha} \left(\sum_{i=1}^{\tau} \sum_{j=1}^{p_i} \alpha_{i,j} x_{i,j} + \sum_{k=1}^{d} \left(\sum_{i=1}^{\tau} \max_{j=1}^{p_i} (w_{i,j}^k - \alpha_{i,j}) + b^k \right) z_k \right) \\ &= \sum_{i=1}^{\tau} \min_{\alpha_{i,1},\ldots,\alpha_{i,p_i}} \left(\sum_{j=1}^{p_i} \alpha_{i,j} x_{i,j} + \sum_{k=1}^{d} z_k \cdot \max_{j=1}^{p_i} (w_{i,j}^k - \alpha_{i,j}) \right) + \sum_{k=1}^{d} b^k z_k. \end{aligned} \tag{15}$$

We show that the minimization problem in (15), for any i, is equivalent to a transportation problem defined as follows.

Definition 3 For any values $x \in \Delta^p$ and $z \in \Delta^d$, and arbitrary weights $w_j^k \in \mathbb{R}$ for all $j \in [\![p]\!]$ and $k \in [\![d]\!]$, define the *max-weight transportation problem* to be

$$\mathsf{Transport}(x, z; w^1, \ldots, w^d) \stackrel{\text{def}}{=} \max_{\beta \geqslant 0} \left\{ \sum_{k=1}^{d} \sum_{j=1}^{p} w_j^k \beta_j^k \;\middle|\; \begin{array}{ll} \sum_{k=1}^{d} \beta_j^k = x_j & \forall j \in [\![p]\!] \\ \sum_{j=1}^{p} \beta_j^k = z_k & \forall k \in [\![d]\!] \end{array} \right\}$$

In the transportation problem, since $\sum_j x_j = 1 = \sum_k z_k$, it follows that $\beta_j^k \in [0, 1]$, and so this value can be interpreted as the percent of total flow "shipped" between j and k. The relation to (15) is now established through LP duality.

Proposition 7 *For any fixed $x \in \Delta^p$ and $z \in \Delta^d$,*

$$\min_{\alpha} \left(\sum_{j=1}^{p} \alpha_j x_j + \sum_{k=1}^{d} z_k \cdot \max_{j=1}^{p} (w_j^k - \alpha_j) \right) = \mathit{Transport}(x, z; w^1, \ldots, w^d). \quad (16)$$

Therefore, when D is a product of simplices, the constraints (12a) can be replaced in (12) with the single inequality

$$y \leq \sum_{i=1}^{\tau} \mathit{Transport}\left((x_{i,j})_{j=1}^{p_i}, z; (w_{i,j}^1)_{j=1}^{p_i}, \ldots, (w_{i,j}^d)_{j=1}^{p_i} \right) + \sum_{k=1}^{d} b^k z_k. \quad (17)$$

Proof By using a variable γ_k to model the value of $\max_{j=1}^{p} (w_j^k - \alpha_j)$ for each $k \in [\![d]\!]$, the minimization problem on the LHS of (16) is equivalent to

$$\min_{\alpha, \gamma} \left\{ \sum_{j=1}^{p} \alpha_j x_j + \sum_{k=1}^{d} \gamma_k z_k \;\middle|\; \gamma_k \geq w_j^k - \alpha_j \quad \forall k \in [\![d]\!],\, j \in [\![p]\!] \right\}$$

which is a minimization LP with free variables α_j and γ_k. Applying LP duality, this completes the proof of Eq. (16). The inequality (17) then arises by substituting Eq. (16) into (15), for every simplex $i = 1, \ldots, \tau$. □

4.3 Simplifications when both $d = 2$ and D is the product of simplices

Proposition 7 shows that, when the input domain is a product of simplices, the tightest upper bound on y can be computed through a series of transportation problems. We now leverage the fact that if either side of the transportation problem from Definition 3 (i.e. p or d) has only two entities, then it reduces to a simpler fractional knapsack

problem. Later, this will allow us to represent (15), in either of the cases $d = 2$ or $p_1 = \cdots = p_\tau = 2$, using an explicit finite family of linear inequalities in x and z which has a greedy linear-time separation oracle.

Proposition 8 *Given data $w^1, w^2 \in \mathbb{R}^p$, take $\widetilde{w}_j = w^1_j - w^2_j$ for all $j \in [\![p]\!]$, and suppose the indices have been sorted so that $\widetilde{w}_1 \le \cdots \le \widetilde{w}_p$. Then*

$$\mathit{Transport}(x, z; w^1, w^2) = \min_{J=1}^{p} \left(\widetilde{w}_J z_1 + \sum_{j=J+1}^{p} (\widetilde{w}_j - \widetilde{w}_J) x_j \right) + \sum_{j=1}^{p} w^2_j x_j. \quad (18)$$

Moreover, a $J \in [\![p]\!]$ that attains the minimum in the right-hand side of (18) can be found in $\mathcal{O}(p)$ time.

Proof When $d = 2$, the transportation problem becomes

$$\max_{\beta^1, \beta^2 \geqslant 0} \left\{ \sum_{j=1}^{p} (w^1_j \beta^1_j + w^2_j \beta^2_j) \;\middle|\; \beta^1_j + \beta^2_j = x_j \quad \forall j \in [\![p]\!], \quad \sum_{j=1}^{p} \beta^1_j = z_1 \right\} \quad (19)$$

where the constraint $\sum_{j=1}^{p} \beta^2_j = z_2$ is implied because $\sum_{j=1}^{p} \beta^2_j = \sum_{j=1}^{p} (x_j - \beta^1_j) = 1 - \sum_{j=1}^{p} \beta^1_j = 1 - z_1 = z_2$. Substituting $\beta^2_j = x_j - \beta^1_j$ for all $j \in [\![p]\!]$ and then omitting the superscript "1", (19) becomes

$$\max_{\beta} \left\{ \sum_{j=1}^{p} (w^1_j - w^2_j) \beta_j + \sum_{j=1}^{p} w^2_j x_j \;\middle|\; \sum_{j=1}^{p} \beta_j = z_1, \quad 0 \le \beta_j \le x_j \quad \forall j \in [\![p]\!] \right\}$$

which is a fractional knapsack problem that can be solved greedily.

An optimal solution to the fractional knapsack LP above, assuming the sorting $w^1_1 - w^2_1 \le \cdots \le w^1_p - w^2_p$, can be computed greedily. Let $J \in [\![p]\!]$ be the *maximum* index at which $\sum_{j=J}^{p} x_j \ge z_1$. We set $\beta_j = 0$ for all $j < J$, $\beta_J = z_1 - \sum_{j=J+1}^{p} x_j$, and $\beta_j = x_j$ for each $j > J$. The optimal cost is

$$\sum_{j=J+1}^{p} (w^1_j - w^2_j) x_j + (w^1_J - w^2_J) \left(z_1 - \sum_{j=J+1}^{p} x_j \right) + \sum_{j=1}^{p} w^2_j x_j,$$

which yields the desired expression after substituting $\widetilde{w}_j = w^1_j - w^2_j$ for all $j \ge J$. Moreover, the index J above can be found in $\mathcal{O}(p)$ time by storing a running total for $\sum_{j=J}^{p} x_j$, completing the proof. □

Observe that the $\mathcal{O}(p)$ runtime in Proposition 8 is non-trivial, as a naïve implementation would run in time $\mathcal{O}(p^2)$, as the inner sum is linear in p.

Combining Propositions 7 and 8 immediately yields the following result.

Corollary 2 *Suppose that $d = 2$ and that D is a product of simplices. Let $z = z_1 \equiv 1 - z_2$. For each simplex $i \in [\![\tau]\!]$, take $\widetilde{w}_{i,j} = w^1_{i,j} - w^2_{i,j}$ for all $j = 1, \ldots, p_i$ and relabel the indices so that $\widetilde{w}_{i,1} \leq \cdots \leq \widetilde{w}_{i,p_i}$. Then, in the context of (12), the upper-bound constraints (12a) are equivalent to*

$$y \leq \sum_{i=1}^{\tau} \left(\widetilde{w}_{i,J(i)} z + \sum_{j=J(i)+1}^{p_i} (\widetilde{w}_{i,j} - \widetilde{w}_{i,J(i)}) x_{i,j} + \sum_{j=1}^{p_i} w^2_{i,j} x_{i,j} \right) + (b^1 - b^2) z + b^2$$
$$\forall \text{ mappings } J : [\![\tau]\!] \to \mathbb{Z} \text{ with } J(i) \in [\![p_i]\!] \; \forall i \in [\![\tau]\!]. \tag{20}$$

Moreover, given any point $(x, y, z) \in D \times \mathbb{R} \times [0, 1]$, feasibility can be verified or a most violated constraint can be found in $\mathcal{O}(p_1 + \cdots + p_\tau)$ time.

Corollary 2 gives an explicit finite family of linear inequalities equivalent to (12). Moreover, we have already shown in Corollary 1 that R_{sharp} yields an ideal formulation when $d = 2$. Hence, we have an ideal nonextended formulation whose exponentially-many inequalities can be separated in $\mathcal{O}(p_1 + \cdots + p_\tau)$ time, where the initial sorting requires $\mathcal{O}(p_1 \log p_1 + \cdots + p_\tau \log p_\tau)$ time. Note that this sorting can be avoided: we may instead solve the fractional knapsack problem in the separation via weighted median in $\mathcal{O}(p_1 + \cdots + p_\tau)$ time [46, Chapter 17.1].

We can also show that none of the constraints in (20) are redundant.

Proposition 9 *Consider the polyhedron P defined as the intersection of all halfspaces corresponding to the inequalities (20). Consider some arbitrary mapping $J : [\![\tau]\!] \to \mathbb{Z}$ with $J(i) \in [\![p_i]\!]$ for each $i \in [\![\tau]\!]$. Then the inequality in (20) for J is irredundant with respect to P. That is, removing the halfspace corresponding to mapping J (note that this halfspace could also correspond to other mappings) from the description of P will strictly enlarge the feasible set.*

Proof Fix a mapping J. Consider the feasible points with $z = 1/2$, and $x_{i,j} = \mathbb{1}[j = J(i)]$ for each i and j. At such points, the constraint corresponding to any mapping $J' \neq J$ in (20) is

$$\begin{aligned} y &\leq \sum_{i=1}^{\tau} \left(\frac{\widetilde{w}_{i,J'(i)}}{2} + \sum_{j=J'(i)+1}^{p_i} (\widetilde{w}_{i,j} - \widetilde{w}_{i,J'(i)}) \mathbb{1}[j = J(i)] + \sum_{j=1}^{p_i} w^2_{i,j} \mathbb{1}[j = J(i)] \right) \\ &\quad + \frac{b^1 + b^2}{2} \\ &= \sum_{i=1}^{\tau} \left(\frac{\widetilde{w}_{i,J'(i)}}{2} + (\widetilde{w}_{i,J(i)} - \widetilde{w}_{i,J'(i)}) \mathbb{1}[J'(i) < J(i)] \right) + \sum_{i=1}^{\tau} w^2_{i,J(i)} + \frac{b^1 + b^2}{2}. \end{aligned}$$

For any simplex i, recall that the indices are sorted so that $\widetilde{w}_{i1} \leq \cdots \leq \widetilde{w}_{ip_i}$. Thus, if $J'(i) \geq J(i)$, then the expression inside the outer parentheses equals $\frac{\widetilde{w}_{i,J'(i)}}{2} \geqslant \frac{\widetilde{w}_{i,J(i)}}{2}$. On the other hand, if $J'(i) < J(i)$, then the expression inside the outer parentheses can be re-written as $\frac{\widetilde{w}_{i,J(i)}}{2} + \frac{\widetilde{w}_{i,J(i)} - \widetilde{w}_{i,J'(i)}}{2} \geqslant \frac{\widetilde{w}_{i,J(i)}}{2}$. Therefore, setting $J'(i) = J(i)$ for every simplex i achieves the tightest upper bound in (20), which simplifies to

$$y \leq \sum_{i=1}^{\tau} \frac{\widetilde{w}_{i,J(i)}}{2} + \sum_{i=1}^{\tau} w^2_{i,J(i)} + \frac{b^1 + b^2}{2}. \tag{21}$$

Now, suppose that the same upper bound on y is achieved by a mapping J' such that $J'(i) \neq J(i)$ on a simplex i. By the argument above, regardless of whether $J'(i) > J(i)$ or $J'(i) < J(i)$, the expression inside the outer parentheses can equal $\frac{\widetilde{w}_{i,J(i)}}{2}$ only if $\widetilde{w}_{i,J'(i)} = \widetilde{w}_{i,J(i)}$. In this case, inspecting the term inside the summation in (20) for mappings J and J', we observe that regardless of the values of x or z,

$$\widetilde{w}_{i,J'(i)} z + \sum_{j=J'(i)+1}^{p_i} (\widetilde{w}_{i,j} - \widetilde{w}_{i,J'(i)}) x_{i,j} = \widetilde{w}_{i,J(i)} z + \sum_{j=J(i)+1}^{p_i} (\widetilde{w}_{i,j} - \widetilde{w}_{i,J(i)}) x_{i,j}.$$

Therefore, for a mapping J' to achieve the tightest upper bound (21), it must be the case that $\widetilde{w}_{i,J'(i)} = \widetilde{w}_{i,J(i)}$ on every simplex i, and so J' and J necessarily correspond to the same half-space in $D \times \mathbb{R} \times [0, 1]$, completing the proof. □

To close this section, we consider the setting where every simplex is 2-dimensional, i.e. $p_1 = \cdots = p_\tau = 2$, but the number of affine functions d can be arbitrary. Note that by contrast, the previous results (Propositions 8, 9, Corollary 2) held in the setting where $d = 2$ but $p_1, \ldots, p_\tau$ were arbitrary. We exploit the symmetry of the transportation problem to immediately obtain the following analogous results, all wrapped up into Corollary 3.

Corollary 3 *Given data $w^1, \ldots, w^d \in \mathbb{R}^2$, take $\widetilde{w}^k = w^k_1 - w^k_2$ for all $k \in [\![d]\!]$, and suppose the indices have been sorted so that $\widetilde{w}^1 \leq \cdots \leq \widetilde{w}^d$. Then*

$$\mathit{Transport}(x, z; w^1, \ldots, w^d) = \min_{K=1}^{d} \left(\widetilde{w}^K x_1 + \sum_{k=1}^{K} w^k_2 z_k + \sum_{k=K+1}^{d} (w^k_1 - \widetilde{w}^K) z_k \right). \tag{22}$$

Moreover, a $K \in [\![d]\!]$ that attains the minimum in the right-hand side of (22) can be found in $\mathcal{O}(d)$ time.

Therefore, suppose that D is a product of τ simplices of dimensions $p_1 = \cdots = p_\tau = 2$. For each simplex $i \in [\![\tau]\!]$, take $\widetilde{w}^k_i = w^k_{i,1} - w^k_{i,2}$ for all $k = 1, \ldots, d$ and relabel the indices so that $\widetilde{w}^1_i \leq \cdots \leq \widetilde{w}^d_i$. Then, in the context of (12), the upper-bound constraints (12a) are equivalent to

$$y \leq \sum_{i=1}^{\tau} \left(\widetilde{w}^{K(i)}_i x_{i,1} + \sum_{k=1}^{K(i)} w^k_{i,2} z_k + \sum_{k=K(i)+1}^{d} (w^k_{i,1} - \widetilde{w}^{K(i)}_i) z_k \right) + \sum_{k=1}^{d} b^k z_k$$
$$\forall \text{ mappings } K : [\![\tau]\!] \to [\![d]\!]. \tag{23}$$

Furthermore, none of the constraints in (23) are redundant.

Corollary 3 will be particularly useful in Sect. 5.1, where it will allow us to derive sharp formulations for the maximum of d affine functions over a box input domain, in analogy to (20).

5 Applications of our machinery

We are now prepared to return to the concrete goal of this paper: building strong MIP formulations for nonlinearities used in modern neural networks.

5.1 A hereditarily sharp formulation for `Max` on box domains

We can now present a hereditarily sharp formulation, with exponentially-many constraints that can be efficiently separated, for the maximum of $d > 2$ affine functions over a shared box input domain.

Proposition 10 *For each $k, \ell \in [\![d]\!]$, take*

$$N^{\ell,k} = \sum_{i=1}^{\eta} \max\{(w_i^k - w_i^\ell)L_i, (w_i^k - w_i^\ell)U_i)\}.$$

A valid MIP formulation for $\mathrm{gr}(\texttt{Max} \circ (f^1, \ldots, f^d); [L, U])$ *is*

$$y \leqslant w^\ell \cdot x + \textstyle\sum_{k=1}^{d} \left(N^{\ell,k} + b^k\right) z_k \quad \forall \ell \in [\![d]\!] \tag{24a}$$

$$y \geqslant w^k \cdot x + b^k \quad \forall k \in [\![d]\!] \tag{24b}$$

$$(x, y, z) \in [L, U] \times \mathbb{R} \times \Delta^d \tag{24c}$$

$$z \in \{0, 1\}^d. \tag{24d}$$

Moreover, a hereditarily sharp formulation is given by (24)*, along with the constraints*

$$y \leqslant \sum_{i=1}^{\eta} \left(w_i^{I(i)} x_i + \sum_{k=1}^{d} \max\{(w_i^k - w_i^{I(i)})L_i, (w_i^k - w_i^{I(i)})U_i\} z_k \right) + \sum_{k=1}^{d} b^k z_k$$
$$\forall \text{ mappings } I : [\![\eta]\!] \to [\![d]\!] \tag{25}$$

Furthermore, none of the constraints (24b) *and* (25) *are redundant.*

Proof The result directly follows from Corollary 3 if we make a change of variables to transform the box domain $[L, U]$ to the domain $(\Delta^2)^\eta$, where the x-coordinates are given by $x_{i,1}, x_{i,2} \geq 0$ over $i \in [\![\eta]\!]$, with $x_{i,1} + x_{i,2} = 1$ for each simplex i. For each i, let $w_{i,1}^k = w_i^k U_i$, $w_{i,2}^k = w_i^k L_i$, and let $\sigma_i : [\![d]\!] \to [\![d]\!]$ be a permutation such that $w_{i,1}^{\sigma_i(1)} - w_{i,2}^{\sigma_i(1)} \leq \cdots \leq w_{i,1}^{\sigma_i(d)} - w_{i,2}^{\sigma_i(d)}$. Fix a mapping $I : [\![\eta]\!] \to [\![d]\!]$

for (23). We make the change of variables $\xi_i \leftarrow (U_i - L_i)x_{i,1} + L_i$ and rewrite (23) from Corollary 3 to take the form (25), as follows:

$$\begin{aligned}
y &\leqslant \sum_{i=1}^{\eta} \left(w_i^{\sigma_i(I(i))}(U_i - L_i)x_{i,1} + \sum_{k=1}^{I(i)} w_i^{\sigma_i(k)} L_i z_k + \sum_{k=I(i)+1}^{d} (w_i^{\sigma_i(k)} U_i - w_i^{\sigma_i(I(i))}(U_i - L_i))z_k \right) \\
&\quad + \sum_{k=1}^{d} b^k z_k, \\
&= \sum_{i=1}^{\eta} \left(w_i^{\sigma_i(I(i))}\xi_i + \sum_{k=1}^{I(i)} (w_i^{\sigma_i(k)} - w_i^{\sigma_i(I(i))}) L_i z_k + \sum_{k=I(i)+1}^{d} (w_i^{\sigma_i(k)} - w_i^{\sigma_i(I(i))}) U_i z_k \right) + \sum_{k=1}^{d} b^k z_k \\
&= \sum_{i=1}^{\eta} \left(w_i^{\sigma_i(I(i))}\xi_i + \sum_{k=1}^{d} \max\{(w_i^{\sigma_i(k)} - w_i^{\sigma_i(I(i))})L_i, (w_i^{\sigma_i(k)} - w_i^{\sigma_i(I(i))})U_i\} z_k \right) + \sum_{k=1}^{d} b^k z_k \\
&= \sum_{i=1}^{\eta} \left(w_i^{\sigma_i(I(i))}\xi_i + \sum_{k=1}^{d} \max\{(w_i^{k} - w_i^{\sigma_i(I(i))})L_i, (w_i^{k} - w_i^{\sigma_i(I(i))})U_i\} z_k \right) + \sum_{k=1}^{d} b^k z_k, \qquad (26)
\end{aligned}$$

where the first equality holds because $\xi_i - (U_i - L_i)x_{i,1} = L_i$ for each i and $\sum_{k=1}^{d} z_k = 1$, the second equality holds because $(w_i^{\sigma_i(k)} - w_i^{\sigma_i(I(i))})L_i \geq (w_i^{\sigma_i(k)} - w_i^{\sigma_i(I(i))})U_i$ if $k \leq I(i)$ (and a reverse argument can be made if $k > I(i)$), and the third equality holds as each $\sigma_i : [\![d]\!] \to [\![d]\!]$ is a bijection. Now, since (26) is taken over all mappings $I : [\![\eta]\!] \to [\![d]\!]$, it is equivalent to replace $\sigma_i(I(i))$ with $I(i)$ in (26). This yields (25) over ξ instead of x.

The lower bound in the context of Corollary 3 can be expressed as $y \geqslant \sum_{i=1}^{\eta}(w_{i,1}^k x_{i,1} + w_{i,2}^k x_{i,2}) + b^k,\ \forall k \in [\![d]\!]$. To transform it to (24b) over ξ, observe that $w_{i,1}^k x_{i,1} + w_{i,2}^k x_{i,2}$ can be re-written as $w_i^k(U_i x_{i,1} + L_i(1 - x_{i,1})) = w_i^k \xi_i$.

Finally, note that this transformation of variables is a bijection since each $x_{i,1}$ was allowed to range over $[0, 1]$, and thus each ξ_i is allowed to range over $[L_i, U_i]$. Hence the new formulation over $(\xi, y, z) \in [L, U] \times \mathbb{R} \times \Delta^d$ is also sharp, yielding the desired result.

The irredundancy of (25) also follows from Corollary 3. For the irredundancy of (24b), fix k and take a point $(\hat{x}, \hat{y}, \hat{z})$ where $f^k(\hat{x}) > f^\ell(\hat{x})$ for all $\ell \neq k$, which exists by Assumption 2. Thus, since the inequalities (24b) are the only ones bounding y from below, the point $(\hat{x}, \hat{y} - \epsilon, \hat{z})$ for some $\epsilon > 0$ is satisfied by all constraints except (24b) corresponding to k, and therefore it is not redundant. □

Observe that the constraints (24a) are a special case of (25) for those constant mappings $I(i) = \ell$ for each $i \in [\![\eta]\!]$. We note in passing that this big-M formulation (24) may be substantially stronger than existing formulations appearing in the literature. For example, the tightest version of the formulation of Tjeng et al. [70] is equivalent to (24) with the coefficients $N^{\ell,k}$ replaced with

$$N^{\ell,k} = b^\ell - b^k + \max_{t \neq \ell} \left(\sum_{i=1}^{\eta} \left(\max\{w_i^t L_i, w_i^t U_i\} - \min\{w_i^\ell L_i, w_i^\ell U_i\} \right) \right).$$

Note in particular that as the inner maximization and minimization are completely decoupled, and that the outer maximization in the definition of $N^{\ell,k,+}$ is completely independent of k.

In "Appendix A", we generalize (24) to provide a valid formulation for arbitrary polytope domains. We next emphasize that the hereditarily sharp formulation from Proposition 10 is particularly strong when $d = 2$.

Corollary 4 *The formulation given by* (24) *and* (25) *is ideal when* $d = 2$. *Moreover, the constraints in the families* (25) *and* (24b) *are facet-defining.*

Proof Idealness follows directly from Corollary 1. Since the constraints (25) and (24b) are irredundant and the formulation is ideal, they must either be facet-defining or describe an implied equality. Given that the equality $\sum_{k=1}^{d} z_k = 1$ appears in (24c), it suffices to observe that the polyhedron defined by (24) and (25) has dimension $\eta + d$, which holds under Assumption 2. □

We can compute a most-violated inequality from the family (25) efficiently.

Proposition 11 *Consider the family of inequalities* (25). *Take some point* $(\hat{x}, \hat{y}, \hat{z}) \in [L, U] \times \mathbb{R} \times \Delta^d$. *If any constraint in the family is violated at the given point, a most-violated constraint can be constructed by selecting* $\hat{I} : [\![\eta]\!] \to [\![d]\!]$ *such that*

$$\hat{I}(i) \in \arg\min_{\ell \in [\![d]\!]} \left(w_i^\ell \hat{x}_i + \sum_{k=1}^{d} \max\{(w_i^k - w_i^\ell)L_i, (w_i^k - w_i^\ell)U_i\}\hat{z}_k \right) \tag{27}$$

for each $i \in [\![\eta]\!]$. *Moreover, if the weights* w_i^k *are sorted on* k *for each* $i \in [\![\eta]\!]$, *this can be done in* $\mathcal{O}(\eta d)$ *time.*

Proof Follows directly from Corollary 3, which says that the minimization problem (27) can be solved in $\mathcal{O}(d)$ time for any $i \in [\![\eta]\!]$. □

Note that naïvely, the minimization problem (27) would take $\mathcal{O}(d^2)$ time, because one has to check every $\ell \in [\![d]\!]$, and then sum over $k \in [\![d]\!]$ for every ℓ. However, if we instead pre-sort the weights $w_i^1, \ldots, w_i^d$ for every $i \in [\![\eta]\!]$ in $\mathcal{O}(\eta d \log d)$ time, we can use Corollary 3 to run efficiently separate via a linear search. We note, however, that this pre-sorting step can potentially be obviated by solving the fractional knapsack problems appearing as a weighted median problem, which can be solved in $\mathcal{O}(d)$ time.

5.2 The ReLU over a box domain

We can now present the results promised in Theorem 2.3. In particular, we derive a non-extended ideal formulation for the ReLU nonlinearity, stated only in terms of the original variables (x, y) and the single additional binary variable z. Put another way, it is the strongest possible tightening that can be applied to the big-M formulation (5), and so matches the strength of the multiple choice formulation without the growth in

the number of variables remarked upon in Sect. 2.2. Notationally, for each $i \in [\![\eta]\!]$ take

$$\breve{L}_i = \begin{cases} L_i & \text{if } w_i \geqslant 0 \\ U_i & \text{if } w_i < 0 \end{cases} \quad \text{and} \quad \breve{U}_i = \begin{cases} U_i & \text{if } w_i \geqslant 0 \\ L_i & \text{if } w_i < 0 \end{cases}.$$

Proposition 12 *Take some affine function $f(x) = w \cdot x + b$ over input domain $D = [L, U]$. The following is an ideal MIP formulation for* $\mathrm{gr}(\texttt{ReLU} \circ f; [L, U])$*:*

$$y \leqslant \sum_{i \in I} w_i (x_i - \breve{L}_i (1 - z)) + \left(b + \sum_{i \notin I} w_i \breve{U}_i \right) z \quad \forall I \subseteq [\![\eta]\!] \tag{28a}$$

$$y \geqslant w \cdot x + b \tag{28b}$$

$$(x, y, z) \in [L, U] \times \mathbb{R}_{\geqslant 0} \times [0, 1] \tag{28c}$$

$$z \in \{0, 1\}. \tag{28d}$$

Furthermore, each inequality in (28a) *and* (28b) *is facet-defining.*

Proof [3]This result is a special case of Corollary 4. Observe that (28) is equivalent to (24) and (25) with $w_i^1 = w_i$ and $w_i^2 = 0$ for all $i \in [\![\eta]\!]$, $b^1 = b$, $b^2 = 0$, $z_1 = z$, and $z_2 = 1 - z$. The constraints (28a) are found by setting $I = \left\{ i \in [\![\eta]\!] \,\middle|\, \hat{I}(i) = 1 \right\}$ for each mapping $\hat{I}$ in (25). □

Formulation (28) has a number of constraints exponential in the input dimension η, so it will not be useful directly as a MIP formulation. However, it is straightforward to separate the exponential family (28a) efficiently.

Proposition 13 *Take $(\hat{x}, \hat{y}, \hat{z}) \in [L, U] \times \mathbb{R}_{\geqslant 0} \times [0, 1]$, along with the set*

$$\hat{I} = \left\{ i \in [\![\eta]\!] \,\middle|\, w_i \hat{x}_i < w_i \left(\breve{L}(1 - \hat{z}) + \breve{U}_i \hat{z} \right) \right\}.$$

If

$$\hat{y} > b\hat{z} + \sum_{i \in \hat{I}} w_i \left(\hat{x}_i - \breve{L}(1 - \hat{z}) \right) + \sum_{i \notin \hat{I}} w_i \breve{U}_i \hat{z},$$

then the constraint in (28a) *corresponding to $\hat{I}$ is the most violated in the family. Otherwise, no inequality in the family is violated at $(\hat{x}, \hat{y}, \hat{z})$.*

Proof Follows as a special case of Proposition 11. □

Note that (5b) and (5c) correspond to (28a) with $I = [\![\eta]\!]$ and $I = \emptyset$, respectively. All this suggests an iterative approach to formulating ReLU neurons over box domains: start with the big-M formulation (5), and use Proposition 13 to separate strengthening inequalities from (28a) as they are needed.

[3] Alternatively, a constructive proof of validity and idealness using Fourier–Motzkin elimination is given in the extended abstract of this work [4, Proposition 1].

5.3 The ReLU with one-hot encodings

Although box domains are a natural choice for many applications, it is often the case that some (or all) of the first layer of a neural network will be constrained to be the product of simplices. The *one-hot encoding* is a standard technique used in the machine learning community to preprocess discrete or categorical data to a format more amenable for learning (see, for example, [16, Chapter 2.2]). More formally, if input x is constrained to take categorical values $x \in C = \{c^1, \ldots, c^t\}$, the one-hot transformation encodes this as $\widetilde{x} \in \{0,1\}^t$, where $\widetilde{x}_i = 1$ if and only if $x = c^i$. In other words, the input is constrained such that $\widetilde{x} \in \underline{\Delta}^\eta \stackrel{\text{def}}{=} \Delta^\eta \cap \{0,1\}^\eta$.

It is straightforward to construct a small ideal formulation for $\text{gr}(\texttt{ReLU} \circ f; \underline{\Delta}^\eta)$ as $\left\{(x, \sum_{i=1}^{\eta} \max\{0, w_i x_i + b\}) \,\middle|\, x \in \underline{\Delta}^\eta\right\}$. However, it is typically the case that multiple features will be present in the input, meaning that the input domain would consist of the product of (potentially many) simplices. For example, neural networks have proven well-suited for predicting the propensity for a given DNA sequence to bind with a given protein [2,83], where the network input consists of a sequence of n base pairs, each of which can take 4 possible values. In this context, the input domain would be $\prod_{i=1}^{n} \underline{\Delta}^4$.

In this section, we restate the general results presented in Sect. 2, specialized for the standard case of the ReLU nonlinearity.

Corollary 5 *Presume that the input domain $D = \Delta^{p_1} \times \cdots \times \Delta^{p_\tau}$ is a product of τ simplices, and that $f(x) = \sum_{i=1}^{\tau} \sum_{j=1}^{p_i} w_{i,j} x_{i,j} + b$ is an affine function. Presume that, for each $i \in [\![\tau]\!]$, the weights are sorted such that $w_{i,1} \leqslant \cdots \leqslant w_{i,p_i}$. Then an ideal formulation for* $\text{gr}(\texttt{ReLU} \circ f; D)$ *is:*

$$y \geqslant w \cdot x + b \tag{29a}$$

$$y \leqslant \sum_{i=1}^{\tau} \left(w_{i,J(i)} z + \sum_{j=J(i)+1}^{p_i} (w_{i,j} - w_{i,J(i)}) x_{i,j} \right) + bz$$

$$\forall \text{ mappings } J : [\![\tau]\!] \to \mathbb{Z} \text{ with } J(i) \in [\![p_i]\!] \; \forall i \in [\![\tau]\!] \tag{29b}$$

$$(x, y, z) \in D \times \mathbb{R}_{\geqslant 0} \times \{0,1\}. \tag{29c}$$

Moreover, a most-violated constraint from the family (29b)*, if one exists, can be identified in $\mathcal{O}(p_1 + \cdots + p_\tau)$ time. Finally, none of the constraints from* (29b) *are redundant.*

Proof Follows directly from applying Corollary 2 to the set R_{sharp}. By Corollary 1, this set actually leads to an ideal formulation, because we are taking the maximum of only two functions (with one of them being zero). The statement about non-redundancy follows from Proposition 9. □

5.4 The leaky ReLU over a box domain

A slightly more exotic variant of the ReLU is the *leaky ReLU*, defined as $\texttt{Leaky}(v; \alpha) = \max\{\alpha v, v\}$ for some constant $0 < \alpha < 1$. Instead of fixing any

negative input to zero, the leaky ReLU scales it by a (typically small) constant α. This has been empirically observed to help avoid the "vanishing gradient" problem during the training of certain networks [55,82]. We present analogous results for the leaky ReLU as for the ReLU: an ideal MIP formulation with an efficient separation routine for the constraints.

Proposition 14 *Take some affine function $f(x) = w \cdot x + b$ over input domain $D = [L, U]$. The following is a valid formulation for* $\text{gr}(\texttt{Leaky} \circ f; [L, U])$*:*

$$y \geqslant f(x) \tag{30a}$$
$$y \geqslant \alpha f(x) \tag{30b}$$
$$y \leqslant f(x) - (1-\alpha) \cdot M^{-}(f; [L, U]) \cdot (1 - z) \tag{30c}$$
$$y \leqslant \alpha f(x) - (\alpha - 1) \cdot M^{+}(f; [L, U]) \cdot z \tag{30d}$$
$$(x, y, z) \in [L, U] \times \mathbb{R} \times [0, 1] \tag{30e}$$
$$z \in \{0, 1\}. \tag{30f}$$

Moreover, an ideal formulation is given by (30)*, along with the constraints*

$$y \leqslant \left(\sum_{i \in I} w_i (x_i - \breve{L}_i (1 - z)) + \left(b + \sum_{i \notin I} w_i \breve{U}_i \right) z \right) + \alpha \left(\sum_{i \notin I} w_i (x_i - \breve{U}_i z) + \left(b + \sum_{i \in I} w_i \breve{L}_i \right) (1 - z) \right) \quad \forall I \subseteq [\![\eta]\!]. \tag{31}$$

Additionally, the most violated inequality from the family (31) *can be separated in $\mathcal{O}(\eta)$ time. Finally, each inequality in* (30a)–(30d) *and* (31) *is facet-defining.*

Proof Follows as a special case of Corollary 4. □

6 Computational experiments

To conclude this work, we perform a preliminary computational study of our approaches for ReLU-based networks. We focus on verifying image classification networks trained on the canonical MNIST digit data set [49]. We train a neural network $f : [0, 1]^{28 \times 28} \to \mathbb{R}^{10}$, where each of the 10 outputs corresponds to the logits[4] for each of the digits from 0 to 9. Given a training image $\tilde{x} \in [0, 1]^{28 \times 28}$, our goal is to prove that there does not exist a perturbation of $\tilde{x}$ such that the neural network f produces a wildly different classification result. If $f(\tilde{x})_i = \max_{j=1}^{10} f(\tilde{x})_j$, then $\tilde{x}$ is placed in class i. Consider an input image with known label i. To evaluate robustness around $\tilde{x}$ with respect to class j, select some small $\epsilon > 0$ and solve the problem $\max_{a : ||a||_\infty \leqslant \epsilon} f(\tilde{x} + a)_j - f(\tilde{x} + a)_i$. If the optimal solution (or a dual bound thereof)

[4] In this context, logits are non-normalized predictions of the neural network so that $\tilde{x} \in [0, 1]^{28 \times 28}$ is predicted to be digit $i - 1$ with probability $\exp(f(\tilde{x})_i) / \sum_{j=1}^{10} \exp(f(\tilde{x})_j)$ [1].

is less than zero, this verifies that our network is robust around $\tilde{x}$ as we cannot produce a small perturbation that will flip the classification from i to j. We note that in the literature there are a number of variants of the verification problem presented above, produced by selecting a different objective function [50,81] or constraint set [28]. We use a model similar to that of Dvijotham et al. [25,26].

We train two models, each using the same architecture, with and without L1 regularization. The architecture starts with a convolutional layer with ReLU activation functions (4 filters, kernel size of 4, stride of 2), which has 676 ReLUs, then a linear convolutional layer (with the same parameters but without ReLUs), feeding into a dense layer of 16 ReLU neurons, and then a dense linear layer with one output per digit representing the logits. Finally, we have a softmax layer that is only enabled during training time to normalize the logits and output probabilities. Such a network is smaller than typically used for image classification tasks, but is nonetheless capable of achieving near-perfect out of sample accuracy on the MNIST data set, and of presenting us with challenging optimization problems. We generate 100 instances for each network by randomly selecting images $\tilde{x}$ with true label i from the test data, along with a random target adversarial class $j \neq i$.

As a general rule of thumb, larger networks will be capable of achieving higher accuracy, but will lead to larger optimization problems which are more difficult to solve. However, even with a fixed network architecture, there can still be dramatic variability in the difficulty of optimizing over different parameter realizations. We refer the interested reader to Ryu et al. [62] for an example of this phenomena in reinforcement learning. Moreover, in the scope of this work we make no attempts to utilize recent techniques that train the networks to be verifiable [25,78,79,81].

For all experiments, we use the Gurobi v7.5.2 solver, running with a single thread on a machine with 128 GB of RAM and 32 CPUs at 2.30 GHz. We use a time limit of 30 minutes (1800 s) for each run. We perform our experiments using the `tf.opt` package for optimization over trained neural networks; `tf.opt` is under active development at Google, with the intention to open source the project in the future. The *big-M + (28a)* method is the big-M formulation (5) paired with separation[5] over the exponential family (28a), and with Gurobi's cutting plane generation turned off. Similarly, the *big-M* and the *extended* methods are the big-M formulation (5) and the extended formulation (7) respectively, with default Gurobi settings. Finally, the *big-M + no cuts* method turns off Gurobi's cutting plane generation without separating over (28a). As a preprocessing step, if we can infer that a neuron is linear based on its bounds (e.g. a nonnegative lower bound or nonpositive upper bound on the input affine function of a ReLU), we transform it into a linear neuron, thus ensuring Assumption 2.

6.1 Network with standard training

We start with a model trained with a standard procedure, using the Adam algorithm [45], running for 15 epochs with a learning rate of 10^{-3}. The model attains

[5] We use cut callbacks in Gurobi to inject separated inequalities into the cut loop. While this offers little control over when the separation procedure is run, it allows us to take advantage of Gurobi's sophisticated cut management implementation.

Table 1 Shifted geometric mean for time and optimality gap taken over 100 instances (shift of 10 and 1, respectively)

Method	Time (s)	Optimality gap (%)	Win
(a) Network with standard training			
Big-M + (28a)	174.49	0.53	81
Big-M	1233.49	6.03	0
Big-M + no cuts	1800.00	125.6	0
Extended	890.21	1.26	6
(b) Network trained with L1 regularization			
Big-M + (28a)	9.17	0	100
Big-M	434.72	1.80	0
Big-M + no cuts	1646.45	21.52	0
Extended	1120.14	3.00	0

The "win" column is the number of (solved) instances on which the method is the fastest

97.2% test accuracy. We select a perturbation ball radius of $\epsilon = 0.1$. We report the results in Table 1a and in Fig. 4a. The big-M + (28a) method solves 7 times faster on average than the big-M formulation. Indeed, for 79 out of 100 instances the big-M method does not prove optimality after 30 minutes, and it is never the fastest choice (the "win" column). Moreover, the big-M + no cuts times out on every instance, implying that using *some* cuts is important. The extended method is roughly 5 times slower than the big-M + (28a) method, but only exceeds the time limit on 19 instances, and so is substantially more reliable than the big-M method for a network of this size.

6.2 ReLU network with L1 regularization

The optimization problems studied in Sect. 6.1 are surprisingly difficult given the relatively small size of the networks involved. This can largely be attributed to the fact that the weights describing the neural network are almost completely dense. To remedy this, we train a second model, using the same network architecture, but with L1 regularization as suggested by Xiao et al. [81]. We again set a radius of $\epsilon = 0.1$, and train for 100 epochs with a learning rate of 5×10^{-4} and a regularization parameter of 10^{-4}. The network achieves a test accuracy of 98.0%. With regularization, 4% of the weights in the network are zero, compared to 0.3% in the previous network. Moreover, with regularization we can infer from variable bounds that on average 73.1% of the ReLUs are linear within the perturbation box of radius ϵ, enabling us to eliminate the corresponding binary variables. In the first network, we can do so only for 27.8% of the ReLUs.

We report the corresponding results in Table 1b and Fig. 4b. While the extended approach does not seem to be substantially affected by the network sparsity, the big-M-based approaches are able to exploit it to solve more instances, more quickly. The big-M approach is able to solve 70 of 100 instances to optimality within the time limit, though the mean solve time is still quite large. In contrast, the big-M + (28a) approach fully exploits the sparsity in the model, solving each instance strictly faster than each

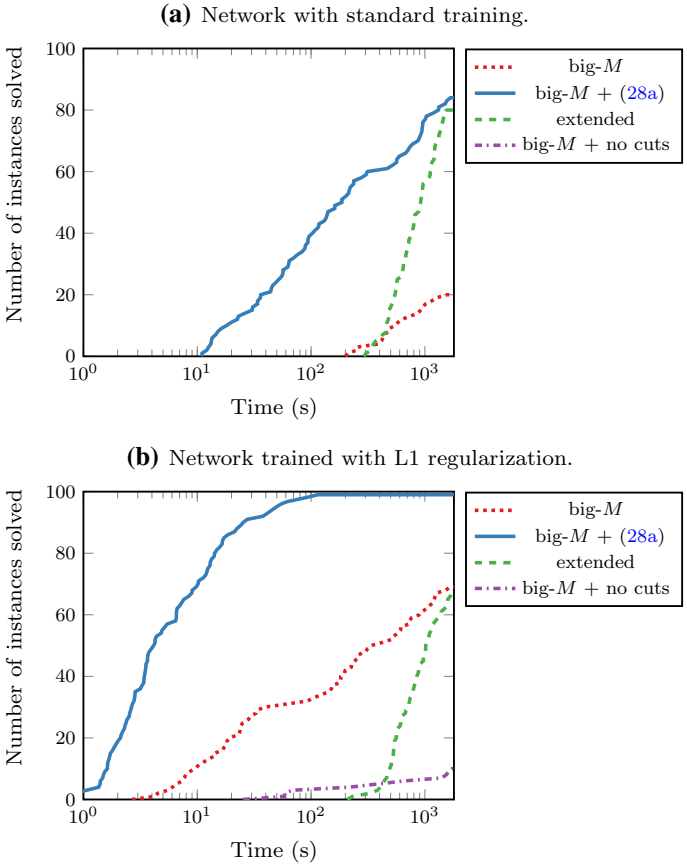


Fig. 4 Number of instances solved within a given amount of time. Curves to the upper left are better, with more instances solved in less time

of the other approaches. Indeed, the approach is able to solve 69 instances in less than 10 s, and solves all instances in under 120 s.

Acknowledgements The authors gratefully acknowledge Yeesian Ng and Ondřej Sýkora for many discussions on the topic of this paper, and for their work on the development of the `tf.opt` package used in the computational experiments.

A A tight big-*M* formulation for `Max` on polyhedral domains

We present a tightened big-*M* formulation for the maximum of d affine functions over an arbitrary polytope input domain. We can view the formulation as a relaxation of the system in Proposition 4, where we select d inequalities from each of (10a) and (10b): those corresponding to $\overline{\alpha}, \underline{\alpha} \in \{w^1, \ldots, w^d\}$. This subset yields a valid formulation, and we obviate the need for direct separation. This formulation can also be viewed as an

application of Proposition 6.2 of Vielma [72], and is similar to the big-M formulations for generalized disjunctive programs of Trespalacios and Grossmann [71].

Proposition 15 *Take coefficients N such that, for each $\ell, k \in [\![d]\!]$ with $\ell \neq k$,*

$$N^{\ell,k,+} \geqslant \max_{x^k \in D_{|k}} \{(w^k - w^\ell) \cdot x^k\} \tag{32a}$$

$$N^{\ell,k,-} \leqslant \min_{x^k \in D_{|k}} \{(w^k - w^\ell) \cdot x^k\}, \tag{32b}$$

and $N^{k,k,+} = N^{k,k,-} = 0$ for all $k \in [\![d]\!]$. Then a valid formulation for $\mathrm{gr}(\texttt{Max} \circ (f^1, \ldots, f^d); D)$ *is:*

$$y \leqslant w^\ell \cdot x + \sum_{k=1}^{d} (N^{\ell,k,+} + b^k) z_k \quad \forall \ell \in [\![d]\!] \tag{33a}$$

$$y \geqslant w^\ell \cdot x + \sum_{k=1}^{d} (N^{\ell,k,-} + b^k) z_k \quad \forall \ell \in [\![d]\!] \tag{33b}$$

$$(x, y, z) \in D \times \mathbb{R} \times \Delta^d \tag{33c}$$

$$z \in \{0, 1\}^d \tag{33d}$$

The tightest possible coefficients in (32) can be computed exactly by solving an LP for each pair of input affine functions $\ell \neq k$. While this might be exceedingly computationally expensive if d is large, it is potentially viable if d is a small fixed constant. For example, the max pooling neuron computes the maximum over a rectangular window in a larger array [32, Sect. 9.3], and is frequently used in image classification architectures. Typically, max pooling units work with a 2×2 or a 3×3 window, in which case $d = 4$ or $d = 9$, respectively.

In addition, if in practice we observe that if the set $D_{|k}$ is empty, then we can infer that the neuron is not irreducible as the k-th input function is never the maximum, and we can safely prune it. In particular, if we attempt to compute the coefficients for z_k and it is proven infeasible, we can prune the k-th function.

References

1. https://developers.google.com/machine-learning/glossary/#logits. Accessed 6 Feb 2020
2. Alipanahi, B., Delong, A., Weirauch, M.T., Frey, B.J.: Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. Nat. Biotechnol. **33**, 831–838 (2015)
3. Amos, B., Xu, L., Kolter, J.Z.: Input convex neural networks. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning, vol. 70, pp. 146–155. PMLR, International Convention Centre, Sydney (2017)
4. Anderson, R., Huchette, J., Tjandraatmadja, C., Vielma, J.P.: Strong mixed-integer programming formulations for trained neural networks. In: A. Lodi, V. Nagarajan (eds.) Proceedings of the 20th Conference on Integer Programming and Combinatorial Optimization, pp. 27–42. Springer International Publishing, Cham (2019). arxiv:1811.08359
5. Arora, R., Basu, A., Mianjy, P., Mukherjee, A.: Understanding deep neural networks with rectified linear units (2016). arXiv preprint arXiv:1611.01491

6. Arulkumaran, K., Deisenroth, M.P., Brundage, M., Bharath, A.A.: Deep reinforcement learning: a brief survey. IEEE Signal Process. Mag. **34**(6), 26–38 (2017)
7. Atamtürk, A., Gómez, A.: Strong formulations for quadratic optimization with M-matrices and indicator variables. Math. Program. **170**, 141–176 (2018)
8. Balas, E.: Disjunctive programming and a hierarchy of relaxations for discrete optimization problems. SIAM J. Algorithmic Discrete Methods **6**(3), 466–486 (1985)
9. Balas, E.: Disjunctive programming: properties of the convex hull of feasible points. Discrete Appl. Math. **89**, 3–44 (1998)
10. Bartolini, A., Lombardi, M., Milano, M., Benini, L.: Neuron constraints to model complex real-world problems. In: International Conference on the Principles and Practice of Constraint Programming, pp. 115–129. Springer, Berlin (2011)
11. Bartolini, A., Lombardi, M., Milano, M., Benini, L.: Optimization and controlled systems: a case study on thermal aware workload dispatching. In: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, pp. 427–433 (2012)
12. Bastani, O., Ioannou, Y., Lampropoulos, L., Vytiniotis, D., Nori, A.V., Criminisi, A.: Measuring neural net robustness with constraints. In: Advances in Neural Information Processing Systems, pp. 2613–2621 (2016)
13. Belotti, P., Bonami, P., Fischetti, M., Lodi, A., Monaci, M., Nogales-Gomez, A., Salvagnin, D.: On handling indicator constraints in mixed integer programming. Comput. Optim. Appl. **65**(3), 545–566 (2016)
14. Bertsimas, D., Tsitsiklis, J.: Introduction to Linear Optimization. Athena Scientific, Belmont, MA (1997)
15. Bienstock, D., Muñoz, G., Pokutta, S.: Principled deep neural network training through linear programming (2018). arXiv preprint arXiv:1810.03218
16. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, Berlin (2006)
17. Bonami, P., Lodi, A., Tramontani, A., Wiese, S.: On mathematical programming with indicator constraints. Math. Program. **151**(1), 191–223 (2015)
18. Boureau, Y.L., Bach, F., LeCun, Y., Ponce, J.: Learning mid-level features for recognition. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 2559–2566 (2010)
19. Bunel, R., Turkaslan, I., Torr, P.H., Kohli, P., Kumar, M.P.: A unified view of piecewise linear neural network verification. In: Advances in Neural Information Processing Systems (2018)
20. Carlini, N., Wagner, D.: Towards evaluating the robustness of neural networks. In: 2017 IEEE Symposium on Security and Privacy (SP), pp. 39–57 (2017)
21. Chen, L., Ma, W., Natarajan, K., Simchi-Levi, D., Yan, Z.: Distributionally robust linear and discrete optimization with marginals. Available at SSRN 3159473 (2018)
22. Cheng, C.H., Nührenberg, G., Ruess, N.: Maximum resilience of artifical neural networks. In: International Symposium on Automated Technology for Verification and Analysis. Springer, Cham (2017)
23. Dulac-Arnold, G., Evans, R., van Hasselt, H., Sunehag, P., Lillicrap, T., Hunt, J., Mann, T., Weber, T., Degris, T., Coppin, B.: Deep reinforcement learning in large discrete action spaces (2015). arxiv:1512.07679
24. Dutta, S., Jha, S., Sanakaranarayanan, S., Tiwari, A.: Output range analysis for deep feedforward neural networks. In: NASA Formal Methods Symposium (2018)
25. Dvijotham, K., Gowal, S., Stanforth, R., Arandjelovic, R., O'Donoghue, B., Uesato, J., Kohli, P.: Training verified learners with learned verifiers (2018). arxiv:1805.10265
26. Dvijotham, K., Stanforth, R., Gowal, S., Mann, T., Kohli, P.: A dual approach to scalable verification of deep networks. In: Thirty-Fourth Conference Annual Conference on Uncertainty in Artificial Intelligence (2018)
27. Ehlers, R.: Formal verification of piece-wise linear feed-forward neural networks. In: International Symposium on Automated Technology for Verification and Analysis. Springer, Cham (2017)
28. Engstrom, L., Tran, B., Tsipras, D., Schmidt, L., Madry, A.: Exploring the landscape of spatial robustness. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning, Proceedings of Machine Learning Research, vol. 97, pp. 1802–1811. PMLR, Long Beach, CA (2019). http://proceedings.mlr.press/v97/engstrom19a.html. Accessed 6 Feb 2020
29. Fischetti, M., Jo, J.: Deep neural networks and mixed integer linear optimization. Constraints **23**, 296–309 (2018)
30. Gatys, L.A., Ecker, A.S., Bethge, M.: A neural algorithm of artistic style (2015). arxiv:1508.06576

31. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, pp. 315–323 (2011)
32. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning, vol. 1. MIT Press, Cambridge (2016)
33. Goodfellow, I.J., Warde-Farley, D., Mirza, M., Courville, A., Bengio, Y.: Maxout networks. In: Proceedings of the 30th International Conference on Machine Learning, vol. 28, pp. 1319–1327 (2013)
34. Grimstad, B., Andersson, H.: ReLU networks as surrogate models in mixed-integer linear programs. Comput. Chem. Eng. **131**, 106580 (2019)
35. Haneveld, W.K.K.: Robustness against dependence in pert: an application of duality and distributions with known marginals. In: Stochastic Programming 84 Part I, pp. 153–182. Springer (1986)
36. Hanin, B.: Universal function approximation by deep neural nets with bounded width and ReLU activations (2017). arXiv preprint arXiv:1708.02691
37. Hijazi, H., Bonami, P., Cornuéjols, G., Ouorou, A.: Mixed-integer nonlinear programs featuring "on/off" constraints. Comput. Optim. Appl. **52**(2), 537–558 (2012)
38. Hijazi, H., Bonami, P., Ouorou, A.: A note on linear on/off constraints (2014). http://www.optimization-online.org/DB_FILE/2014/04/4309.pdf. Accessed 6 Feb 2020
39. Huber, B., Rambau, J., Santos, F.: The Cayley Trick, lifting subdivisions and the Bohne-Dress theorem of zonotopal tiltings. J. Eur. Math. Soc. **2**(2), 179–198 (2000)
40. Huchette, J.: Advanced mixed-integer programming formulations: methodology, computation, and application. Ph.D. thesis, Massachusetts Institute of Technology (2018)
41. Jeroslow, R., Lowe, J.: Modelling with integer variables. Math. Program. Study **22**, 167–184 (1984)
42. Jeroslow, R.G.: Alternative formulations of mixed integer programs. Ann. Oper. Res. **12**, 241–276 (1988)
43. Katz, G., Barrett, C., Dill, D.L., Julian, K., Kochenderfer, M.J.: Reluplex: an efficient SMT solver for verifying deep neural networks. In: International Conference on Computer Aided Verification, pp. 97–117 (2017)
44. Khalil, E.B., Gupta, A., Dilkina, B.: Combinatorial attacks on binarized neural networks. In: International Conference on Learning Representations (2019)
45. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2014). arxiv:1412.6980
46. Korte, B., Vygen, J.: Combinatorial Optimization: Theory and Algorithms. Springer, Berlin (2000)
47. Kumar, A., Serra, T., Ramalingam, S.: Equivalent and approximate transformations of deep neural networks (2019). arxiv:1905.11428
48. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015)
49. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**, 2278–2324 (1998)
50. Liu, C., Arnon, T., Lazarus, C., Barrett, C., Kochenderfer, M.J.: Algorithms for verifying deep neural networks (2019). arxiv:1903.06758
51. Lombardi, M., Gualandi, S.: A lagrangian propagator for artificial neural networks in constraint programming. Constraints **21**(4), 435–462 (2016)
52. Lombardi, M., Milano, M.: Boosting combinatorial problem modeling with machine learning. In: Proceedings IJCAI, pp. 5472–5478 (2018)
53. Lombardi, M., Milano, M., Bartolini, A.: Empirical decision model learning. Artif. Intell. **244**, 343–367 (2017)
54. Lomuscio, A., Maganti, L.: An approach to reachability analysis for feed-forward ReLU neural networks (2017). arxiv:1706.07351
55. Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models. In: ICML Workshop on Deep Learning for Audio, Speech and Language (2013)
56. Mladenov, M., Boutilier, C., Schuurmans, D., Elidan, G., Meshi, O., Lu, T.: Approximate linear programming for logistic Markov decision processes. In: Proceedings of the Twenty-sixth International Joint Conference on Artificial Intelligence (IJCAI-17), pp. 2486–2493. Melbourne, Australia (2017)
57. Mordvintsev, A., Olah, C., Tyka, M.: Inceptionism: Going deeper into neural networks (2015). https://ai.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html. Accessed 6 Feb 2020
58. Natarajan, K., Song, M., Teo, C.P.: Persistency model and its applications in choice modeling. Manage. Sci. **55**(3), 453–469 (2009)
59. Olah, C., Mordvintsev, A., Schubert, L.: Feature visualization. Distill (2017). https://distill.pub/2017/feature-visualization. Accessed 6 Feb 2020

60. Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z.B., Swami, A.: The limitations of deep learning in adversarial settings. In: IEEE European Symposium on Security and Privacy, pp. 372–387 (2016)
61. Raghunathan, A., Steinhardt, J., Liang, P.: Semidefinite relaxations for certifying robustness to adversarial examples. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems, NIPS'18, pp. 10,900–10,910. Curran Associates Inc. (2018)
62. Ryu, M., Chow, Y., Anderson, R., Tjandraatmadja, C., Boutilier, C.: CAQL: Continuous action Q-learning (2019). arxiv:1909.12397
63. Salman, H., Yang, G., Zhang, H., Hsieh, C.J., Zhang, P.: A convex relaxation barrier to tight robustness verification of neural networks (2019). arxiv:1902.08722
64. Say, B., Wu, G., Zhou, Y.Q., Sanner, S.: Nonlinear hybrid planning with deep net learned transition models and mixed-integer linear programming. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17, pp. 750–756 (2017)
65. Schweidtmann, A.M., Mitsos, A.: Global deterministic optimization with artificial neural networks embedded. J. Optim. Theory Appl. **180**, 925–948 (2019)
66. Serra, T., Ramalingam, S.: Empirical bounds on linear regions of deep rectifier networks (2018). arxiv:1810.03370
67. Serra, T., Tjandraatmadja, C., Ramalingam, S.: Bounding and counting linear regions of deep neural networks. In: Thirty-Fifth International Conference on Machine Learning (2018)
68. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing properties of neural networks. In: International Conference on Learning Representations (2014)
69. Tawarmalani, M., Sahinidis, N.: Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming: Theory, Algorithms, Software and Applications, vol. 65. Springer, Berlin (2002)
70. Tjeng, V., Xiao, K., Tedrake, R.: Verifying neural networks with mixed integer programming. In: International Conference on Learning Representations (2019)
71. Trespalacios, F., Grossmann, I.E.: Improved big-M reformulation for generalized disjunctive programs. Comput. Chem. Eng. **76**, 98–103 (2015)
72. Vielma, J.P.: Mixed integer linear programming formulation techniques. SIAM Rev. **57**(1), 3–57 (2015)
73. Vielma, J.P.: Embedding formulations and complexity for unions of polyhedra. Manage. Sci. **64**(10), 4471–4965 (2018)
74. Vielma, J.P.: Small and strong formulations for unions of convex sets from the Cayley embedding. Math. Program. **177**, 21–53 (2018)
75. Vielma, J.P., Nemhauser, G.: Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. Math. Program. **128**(1–2), 49–72 (2011)
76. Weibel, C.: Minkowski sums of polytopes: combinatorics and computation. Ph.D. thesis, École Polytechnique Fédérale de Lausanne (2007)
77. Weiss, G.: Stochastic bounds on distributions of optimal value functions with applications to pert, network flows and reliability. Oper. Res. **34**(4), 595–605 (1986)
78. Wong, E., Kolter, J.Z.: Provable defenses against adversarial examples via the convex outer adversarial polytope. In: International Conference on Machine Learning (2018)
79. Wong, E., Schmidt, F., Metzen, J.H., Kolter, J.Z.: Scaling provable adversarial defenses. In: 32nd Conference on Neural Information Processing Systems (2018)
80. Wu, G., Say, B., Sanner, S.: Scalable planning with Tensorflow for hybrid nonlinear domains. In: Advances in Neural Information Processing Systems, pp. 6276–6286 (2017)
81. Xiao, K.Y., Tjeng, V., Shafiullah, N.M., Madry, A.: Training for faster adversarial robustness verification via inducing ReLU stability. In: International Conference on Learning Representations (2019)
82. Xu, B., Wang, N., Chen, T., Li, M.: Empirical evaluation of rectified activations in convolution network (2015). arxiv:1505.00853
83. Zeng, H., Edwards, M.D., Liu, G., Gifford, D.K.: Convolutional neural network architectures for predicting DNA-protein binding. Bioinformatics **32**(12), 121–127 (2016)