

On Integer Programming for the Binarized Neural Network Verification Problem

Woojin Kim, James Luedtke

October 1, 2025

Abstract

Binarized neural networks (BNNs) are feedforward neural networks with binary weights and activation functions. In the context of using a BNN for classification, the verification problem seeks to determine whether a small perturbation of a given input can lead it to be misclassified by the BNN, and the robustness of the BNN can be measured by solving the verification problem over multiple inputs. The BNN verification problem can be formulated as an integer programming (IP) problem. However, the natural IP formulation is often challenging to solve due to a large integrality gap induced by big- M constraints. We present two techniques to improve the IP formulation. First, we introduce a new method for obtaining a linear objective for the multi-class setting. Second, we introduce a new technique for generating valid inequalities for the IP formulation that exploits the recursive structure of BNNs. We find that our techniques enable verifying BNNs against a higher range of input perturbation than existing IP approaches within a limited time.

1 Introduction

Binarized neural networks (BNNs) are feedforward neural networks with binary weights and activation functions (Hubara et al. (2016)). BNNs consist of the input layer $\ell = 0$, L hidden layers $\ell = 1, \dots, L$, and the output layer $\ell = L + 1$. Each layer $\ell \in \{0, 1, \dots, L + 1\}$ consists of n^ℓ neurons, $N^\ell = \{1, \dots, n^\ell\}$. Every $i \in N^\ell$ is connected to every $j \in N^{\ell-1}$ with weight $W_{ij}^\ell \in \{-1, 0, 1\}$ and has bias $b_i^\ell \in \mathbb{Z}$ for $\ell \in \{1, \dots, L + 1\}$.

Thanks to binary weights and activation functions, BNNs drastically reduce memory size and accesses and substantially improve power-efficiency over standard neural networks, replacing most arithmetic operations with bit-wise operations (Hubara et al. (2016)). BNNs have also achieved nearly state-of-the-art results in image classification (Hubara et al. (2016)). Moreover, BNNs have achieved results comparable to feedforward neural networks in image detection (Kung et al. (2018)), image super resolution (Ma et al. (2019)), and text classification (Shridhar et al. (2020)). For these reasons, BNNs have been applied in small embedded devices (McDanel et al. (2017)).

In this work, we study the verification problem associated with a given BNN that is used to classify feature vectors. BNNs map a feature vector in $[0, 1]^{n^0}$ whose coordinates are quantized as multiples of $\frac{1}{q}$ for $q \in \mathbb{N}$ to the real output vector by the function $f = (f_1, \dots, f_{n^{L+1}}) : \frac{1}{q}\mathbb{Z}_+^{n^0} \cap [0, 1]^{n^0} \rightarrow \mathbb{R}^{n^{L+1}}$ which is defined recursively using weights and biases in each layer. (See Section 2 for the detailed definition of f .) Each $t \in N^{L+1}$ corresponds to a class. A feature vector $\bar{\mathbf{x}} \in \frac{1}{q}\mathbb{Z}_+^{n^0} \cap [0, 1]^{n^0}$ is classified as the class $\bar{t} \in N^{L+1}$ that corresponds to $\arg \max_{t \in N^{L+1}} f_t(\bar{\mathbf{x}})$.

Verifying BNNs against input perturbation for multiple feature vectors provides a measure of their robustness. The BNN verification problem is defined for a given trained BNN, a given feature

vector $\bar{\mathbf{x}}$ with class \bar{t} , and input perturbation $\epsilon > 0$. We say that $\bar{\mathbf{x}}$ is ϵ -verified if the BNN classifies \mathbf{x}^0 as \bar{t} for every feature vector $\mathbf{x}^0 \in \frac{1}{q}\mathbb{Z}_+^{n^0} \cap [0, 1]^{n^0}$ having distance between \mathbf{x}^0 and $\bar{\mathbf{x}}$ at most ϵ . The BNN verification problem is to determine whether a given $\bar{\mathbf{x}}$ can be ϵ -verified in the BNN. The answer is true if and only if there does not exist a feature vector \mathbf{x}^0 with distance at most ϵ from $\bar{\mathbf{x}}$ satisfying $f_t(\mathbf{x}^0) > f_{\bar{t}}(\mathbf{x}^0)$ for some class $t \neq \bar{t}$, which is equivalent to the following maximum $z_\epsilon^*(\bar{\mathbf{x}})$ being non-positive:

$$z_\epsilon^*(\bar{\mathbf{x}}) = \max_{\substack{\mathbf{x}^0 \in \frac{1}{q}\mathbb{Z}_+^{n^0} \cap [0, 1]^{n^0} \\ \|\mathbf{x}^0 - \bar{\mathbf{x}}\|_p \leq \epsilon}} \left\{ f_t(\mathbf{x}^0) - f_{\bar{t}}(\mathbf{x}^0) : t \in N^{L+1} \setminus \{\bar{t}\} \right\}. \quad (1)$$

Here, the maximum perturbation from $\bar{\mathbf{x}}$ is defined using an ℓ_p norm. For the most part, in this work we focus on the case of $p = 1$, but the integer programming (IP) approaches we consider can also be directly applied to the cases of $p = \infty$ and $p = 2$, with the caveat that when using an ℓ_2 norm, the resulting IP formulations we consider become integer (convex) quadratic programs, as opposed to integer linear programs.

This paper contributes to the literature exploring IP methods to solve the BNN verification problem. IP methods to solve the BNN verification problem can be implemented by solving (1) to optimality – we refer to this as the BNN verification optimization problem. However, to verify a BNN, the process of solving (1) can be terminated as soon as the sign of $z_\epsilon^*(\bar{\mathbf{x}})$ is determined. Specifically, the solution process of (1) can be terminated when a feasible solution with a positive objective value is found, or a non-positive upper bound for the optimal objective value is obtained. If we terminate an optimization algorithm for solving (1) according to this condition, we refer to this as the BNN verification problem.

Our first contribution is to present a new way for obtaining a linear objective of the BNN verification problem by creating a single optimization problem that incorporates the decision of which alternative class a perturbed feature vector is misclassified as, rather than considering each alternative class separately. Our second contribution is to describe a new technique for generating valid inequalities for the IP formulation by exploiting the recursive structure of BNNs. These valid inequalities, called layerwise derived valid inequalities, are generated by considering one layer at a time and solving IP subproblems to check validity among a set of natural candidate inequalities. While deriving these valid inequalities requires solving IP subproblems, these IP subproblems are much easier to solve than the full BNN verification problem because they consider only a single layer at a time and do not include any of the “big- M ” constraints required for the IP formulation of the BNN verification problem.

Khalil et al. (2019) study the problem of attacking a BNN, which is equivalent to the BNN verification problem. They present a mixed-integer linear programming (MILP) formulation of the problem and propose a heuristic for generating solutions that works by propagating a target from the last layer to the first layer by solving a single-layer MILP at each step. This approach is proven to be successful at finding successful attacks (i.e., showing an input $\bar{\mathbf{x}}$ cannot be ϵ -verified) but cannot positively verify a solution. Interestingly, our method also proceeds in layers, but from the initial layer forwards, and its strength is in being able to ϵ -verify inputs. Han and Gómez (2021) also study the MILP formulation of the BNN verification problem and study the convex hull for a single neuron to improve this MILP formulation. Lubczyk and Neto (2024) extended Han and Gómez’s work by exploring the convex hull for a pair of neurons in the same hidden layer. Using this convex hull, they proposed a constraint generation framework to solve the MILP problem for the BNN verification problem. In related work on feedforward (not binarized) neural networks, Fischetti and Jo (2018) addressed the problem of verifying such networks. They formulated this problem as an MILP problem and exploited the idea of fixing variables to solve this MILP problem.

Our layerwise derived valid inequalities represent an adaptation and extension of this approach to the BNN setting.

Another approach for solving the BNN verification problem is to formulate it as a Boolean satisfiability problem (Amir et al. (2021), Jia and Rinard (2020), Kovásznaı et al. (2021), Narodytka et al. (2018)). These papers formulated the condition that $\bar{\mathbf{x}}$ can be ϵ -verified in a given BNN with Boolean formulas in the case that the maximum perturbation from $\bar{\mathbf{x}}$ is defined using an ℓ_∞ norm. Ivashchenko et al. (2023) explored a different method for solving the BNN verification problem by describing the set of all feasible output vectors or an overapproximation of this set as a star set and checking whether this star set contains an output vector from a perturbed feature vector misclassified by the BNN. This work also handles the case that the maximum perturbation is defined using an ℓ_∞ norm. Shih et al. (2019) and Zhang et al. (2021) addressed a different problem of *counting* the number of perturbed feature vectors misclassified by the BNN in the case that the maximum perturbation is defined using an ℓ_1 norm. We focus on the IP approach due to its inherent flexibility, for example, in easily adapting the set of allowed perturbations from $\bar{\mathbf{x}}$ to be defined by an ℓ_1 , ℓ_2 , or ℓ_∞ norm, and to investigate and extend the limits of the IP approach.

While our work focuses on verifying a given BNN after it has been trained, for completeness, we briefly mention some methods for training BNNs. Hubara et al. (2016) trained BNNs by using a gradient descent method similar to the training of feedforward neural networks, and Geiger and Team (2020) trained BNNs by using a pseudo-gradient method. Bernardelli et al. (2023) and Toro Icarte et al. (2019) trained BNNs by solving constraint programming or mixed integer programming problems. To improve training BNNs, Martinez et al. (2020) applied activation scaling, and Tang et al. (2017) applied activation approximation.

Finally, we mention that the BNN verification problem is closely related to the problem of finding counterfactual explanations for such networks. Counterfactual explanations seek to find a feature vector that is as close as possible to a given feature vector while leading to a specific alternative desired prediction (Mothilal et al. (2020), Wachter et al. (2018)). Recently, counterfactual explanations for several machine learning prediction models, including k-nearest neighbors, have been explored (Contardo et al. (2024), Vivier-Ardissou et al. (2024)). A method for solving the BNN verification problem can be used to solve the counterfactual explanation problem by conducting a binary search to (approximately) find the smallest ϵ such that the feature is classified as a target class.

This paper is organized as follows. In Section 2, we introduce an existing IP formulation of the BNN verification problem and describe methods for obtaining linear constraints and a linear objective in this IP formulation, including our method for obtaining a linear objective by incorporating the decision of which alternative class a perturbed feature vector is misclassified into a single optimization problem. In Section 3, we describe our technique for deriving layerwise derived valid inequalities and our approach for using these valid inequalities to solve the BNN verification problem. We present the results of a computational study in Section 4 and conclude with a discussion of potential future directions in Section 5.

Notation. We use $[M]$ to represent the set $\{1, \dots, M\}$ for $M \in \mathbb{N}$.

2 IP Formulations

To obtain an IP formulation for problem (1), $\mathbf{x}^0 \in \frac{1}{q}\mathbb{Z}_+^{n^0} \cap [0, 1]^{n^0}$ is defined as a vector of the decision variables that represent a perturbed feature vector, and $\mathbf{x}^\ell \in \{0, 1\}^{n^\ell}$ is defined as the vector of decision variables that represent the output of the ℓ^{th} hidden layer for $\ell \in [L]$, which is obtained recursively from $\mathbf{x}^{\ell-1}$ as explained in the following paragraphs. Also, X^0 is defined as the

set of all perturbed feature vectors to verify over:

$$X^0 = \left\{ \mathbf{x}^0 \in \frac{1}{q} \mathbb{Z}_+^{n^0} \cap [0, 1]^{n^0} : \|\mathbf{x}^0 - \bar{\mathbf{x}}\|_p \leq \epsilon \right\}.$$

The usual description of a BNN describes the output of each neuron as +1 or -1. In order to derive an IP formulation that can be directly solved by IP solvers, we will instead encode the outputs with binary (0/1) valued variables. Given a binary output vector $\mathbf{x}^{\ell-1}$, it can be converted to a +1/-1 valued vector via the transformation $2\mathbf{x}^{\ell-1} - \mathbf{1}$. Thus, for each layer $\ell \in [L+1]$, the first component of BNN propagation is to multiply the input $2\mathbf{x}^{\ell-1} - \mathbf{1}$ with \mathbf{W}^ℓ and add \mathbf{b}^ℓ . We introduce the notation $a^\ell = (a_1^\ell, \dots, a_{n^\ell}^\ell)$ for this affine function of $\mathbf{x}^{\ell-1}$:

$$a^\ell(\mathbf{x}^{\ell-1}) = \mathbf{W}^\ell(2\mathbf{x}^{\ell-1} - \mathbf{1}) + \mathbf{b}^\ell.$$

Then, the output of each each $\ell \in [L]$ is obtained from its input $\mathbf{x}^{\ell-1}$ via the transformation

$$\mathbf{x}^\ell = \mathbf{1}_{\mathbb{R}_+}(a^\ell(\mathbf{x}^{\ell-1})),$$

where $\mathbf{1}_{\mathbb{R}_+}$ is the indicator function mapping non-negative numbers to 1 and negative numbers to 0.¹

Applying a^ℓ and $\mathbf{1}_{\mathbb{R}_+}$ alternately L times and a^{L+1} lastly results in f . Thus, f can be written as follows:

$$f(\mathbf{x}^0) = a^{L+1}(\mathbf{1}_{\mathbb{R}_+}(a^L(\mathbf{1}_{\mathbb{R}_+}(a^{L-1}(\dots \mathbf{1}_{\mathbb{R}_+}(a^1(\mathbf{x}^0)) \dots)))).$$

Using these observations, (1) can be written as the following (nonlinear) IP problem:

$$\max_{\mathbf{x}^0, \dots, \mathbf{x}^L} \quad \max \left\{ a_t^{L+1}(\mathbf{x}^L) - a_{\bar{t}}^{L+1}(\mathbf{x}^L) : t \in N^{L+1} \setminus \{\bar{t}\} \right\} \quad (2a)$$

$$\text{s.t.} \quad \mathbf{x}^\ell = \mathbf{1}_{\mathbb{R}_+}(a^\ell(\mathbf{x}^{\ell-1})), \quad \forall \ell \in [L], \quad (2b)$$

$$\mathbf{x}^0 \in X^0, \quad (2c)$$

$$\mathbf{x}^\ell \in \{0, 1\}^{n^\ell}, \quad \forall \ell \in [L]. \quad (2d)$$

2.1 Constraint Formulation

The layer propagation constraints (2b) and the input perturbation constraint (2c) include nonlinear constraints in their description. We review how these constraints can be reformulated using linear constraints in order to obtain a linear IP formulation.

2.1.1 Layer Propagation Constraint Formulation

To formulate the layer propagation constraints (2b) with linear constraints, the following constants are defined for $\ell \in [L]$ and $i \in N^\ell$:

$$\begin{aligned} \text{LB}_i^\ell &:= \sum_{j \in N^{\ell-1}} (W_{ij}^\ell - |W_{ij}^\ell|) \\ \text{UB}_i^\ell &:= \sum_{j \in N^{\ell-1}} (W_{ij}^\ell + |W_{ij}^\ell|) \\ R_i^\ell &:= \begin{cases} \frac{2}{q} \left\lceil \frac{q(\sum_{j \in N^0} W_{ij}^1 - b_i^1)}{2} \right\rceil - \frac{1}{q} & (\ell = 1) \\ 2 \left\lceil \frac{\sum_{j \in N^{\ell-1}} W_{ij}^\ell - b_i^\ell}{2} \right\rceil - 1 & (\ell \in \{2, \dots, L\}). \end{cases} \end{aligned} \quad (3)$$

¹This is equivalent to the standard description of BNN propagation in which the +1/-1 output is defined by the sign function.

Since $W_{ij}^\ell \in \{-1, 0, 1\}$ it follows that for any $\mathbf{x}^{\ell-1} \in [0, 1]^{n^{\ell-1}}$

$$\text{LB}_i^\ell \leq 2 \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1} \leq \text{UB}_i^\ell.$$

The following lemma describes how the layer propagation constraints can be modeled with linear constraints. This extends the formulation of Han and Gómez (2021), which assumes $W_{ij}^\ell \in \{-1, 1\}$ and $b_i^\ell = 0$. The proof is in the Appendix.

Lemma 1. *Each $(\mathbf{x}^0, \mathbf{x}^1) \in X^0 \times \{0, 1\}^{n^1}$ satisfies (2b) for $\ell = 1$ if and only if it satisfies the following inequalities:*

$$2 \sum_{j \in N^0} W_{ij}^1 x_j^0 \geq \left(R_i^1 - \text{LB}_i^1 + \frac{1}{q}\right) x_i^1 + \text{LB}_i^1, \quad \forall i \in N^1, \quad (4)$$

$$2 \sum_{j \in N^0} W_{ij}^1 x_j^0 \leq \left(\text{UB}_i^1 - R_i^1 + \frac{1}{q}\right) x_i^1 + \left(R_i^1 - \frac{1}{q}\right), \quad \forall i \in N^1. \quad (5)$$

For $\ell \in \{2, \dots, L\}$, $(\mathbf{x}^{\ell-1}, \mathbf{x}^\ell) \in \{0, 1\}^{n^{\ell-1}} \times \{0, 1\}^{n^\ell}$ satisfies (2b) if and only if it satisfies the following inequalities:

$$2 \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1} \geq (R_i^\ell - \text{LB}_i^\ell + 1) x_i^\ell + \text{LB}_i^\ell, \quad \forall i \in N^\ell, \quad (6)$$

$$2 \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1} \leq (\text{UB}_i^\ell - R_i^\ell + 1) x_i^\ell + (R_i^\ell - 1), \quad \forall i \in N^\ell. \quad (7)$$

Thus, the nonlinear constraints (2b) can be replaced with the linear constraints (4)- (7).

2.1.2 Input Perturbation Constraint Formulation

The input perturbation constraint (2c) can be formulated by defining the integer-valued decision variable y_i to represent qx_i^0 , for $i \in N^0$. Then, $\mathbf{x}^0 \in X^0$ if and only if $\mathbf{x}^0 \in \mathbb{R}_+^{n^0}$, and there exist $\mathbf{y} \in \mathbb{Z}_+^{n^0}$ satisfying the following constraints:

$$\mathbf{y} = q\mathbf{x}^0, \quad (8)$$

$$\mathbf{y} \leq q\mathbf{1}, \quad (9)$$

$$\|\mathbf{x}^0 - \bar{\mathbf{x}}\|_p \leq \epsilon. \quad (10)$$

In the case of $p = 1$ or $p = \infty$, (10) can be formulated with linear constraints using standard techniques. In the case of $p = 2$, (10) can be formulated as a convex quadratic constraint.

2.2 Objective Formulation

Finally, to obtain a linear IP formulation of (2) we explore two methods to obtain a linear objective.

The first approach, which has been used in Han and Gómez (2021), Lubczyk and Neto (2024), is to consider separately each alternative class $t \in N^{L+1} \setminus \{\bar{t}\}$. Each such class t results in the

following IP problem with linear objective:

$$\begin{aligned}
z_\epsilon^*(\bar{\mathbf{x}}, t) = \max \quad & a_t^{L+1}(\mathbf{x}^L) - a_{\bar{t}}^{L+1}(\mathbf{x}^L) \\
\text{s.t.} \quad & (4) - (5), \\
& (6) - (7), \quad \forall \ell \in \{2, \dots, L\}, \\
& (8) - (10), \\
& \mathbf{x}^0 \in \mathbb{R}_+^{n^0}, \\
& \mathbf{x}^\ell \in \{0, 1\}^{n^\ell}, \quad \forall \ell \in [L], \\
& \mathbf{y} \in \mathbb{Z}_+^{n^0}.
\end{aligned} \tag{11}$$

The optimal solution of (2) can be obtained by solving (11) for each $t \in N^{L+1} \setminus \{\bar{t}\}$ and choosing t that achieves the maximum of $z_\epsilon^*(\bar{\mathbf{x}}, t)$.

For problems with more than two classes, we propose an alternative approach for obtaining a linear objective which directly includes the choice of the alternative class t in the formulation, thereby avoiding the need to solve (11) for each $t \in N^{L+1} \setminus \{\bar{t}\}$. Thus, for each $t \in N^{L+1} \setminus \{\bar{t}\}$ let $z_t \in \{0, 1\}$ be a decision variable indicating whether t is selected as the alternative class, and for each $i \in N^L$ let $v_{ti} \in \{0, 1\}$ be a decision variable that is used to represent the product $z_t x_i^L$. The proof of the following lemma is in the appendix.

Lemma 2. *The following IP problem is equivalent to (2):*

$$\begin{aligned}
\max \quad & \sum_{t \in N^{L+1} \setminus \{\bar{t}\}} \sum_{i \in N^L} 2(W_{ti}^L - W_{\bar{t}i}^L)v_{ti} \\
& + \sum_{t \in N^{L+1} \setminus \{\bar{t}\}} \left(- \sum_{i \in N^L} (W_{ti}^L - W_{\bar{t}i}^L) + (b_t^L - b_{\bar{t}}^L) \right) z_t
\end{aligned} \tag{12a}$$

$$\begin{aligned}
\text{s.t.} \quad & (4) - (5), \\
& (6) - (7), \quad \forall \ell \in \{2, \dots, L\}, \\
& (8) - (10),
\end{aligned}$$

$$\sum_{t \in N^{L+1} \setminus \{\bar{t}\}} z_t = 1, \tag{12b}$$

$$\sum_{t \in N^{L+1} \setminus \{\bar{t}\}} v_{ti} = x_i^L, \quad \forall i \in N^L, \tag{12c}$$

$$v_{ti} \leq z_t, \quad \forall t \in N^{L+1} \setminus \{\bar{t}\}, i \in N^L, \tag{12d}$$

$$\mathbf{x}^0 \in \mathbb{R}_+^{n^0}, \tag{12e}$$

$$\mathbf{x}^\ell \in \{0, 1\}^{n^\ell}, \quad \forall \ell \in [L], \tag{12f}$$

$$\mathbf{y} \in \mathbb{Z}_+^{n^0}, \tag{12g}$$

$$z_t \in \{0, 1\}, \quad \forall t \in N^{L+1} \setminus \{\bar{t}\}, \tag{12h}$$

$$v_{ti} \in \{0, 1\}, \quad \forall t \in N^{L+1} \setminus \{\bar{t}\}, i \in N^L. \tag{12i}$$

2.3 Valid Inequalities for Neurons

Han and Gómez (2021) explored the following convex hull for a single neuron for $\ell \in \{2, \dots, L\}$

and $i \in N^\ell$ in the case of $\mathbf{b}^\ell = \mathbf{0}$:

$$\text{conv}(\{(\mathbf{x}^{\ell-1}, x_i^\ell) \in \{0, 1\}^{n^{\ell-1}} \times \{0, 1\} : x_i^\ell = \mathbf{1}_{\mathbb{R}_+}(a_i^\ell(\mathbf{x}^{\ell-1}))\}). \quad (13)$$

In the following theorem, we extend the results in Han and Gómez (2021) to a general case where \mathbf{b}^ℓ are not necessarily zero. This theorem is proved in the Appendix.

Theorem 3. For $\ell \in \{2, \dots, L\}$ and $i \in N^\ell$, (13) is the set of $(\mathbf{x}^{\ell-1}, x_i^\ell) \in [0, 1]^{n^{\ell-1}} \times [0, 1]$ satisfying the following inequalities for $J \subset \{j \in N^{\ell-1} : W_{ij}^\ell \neq 0\}$:

$$\sum_{j \in J} (W_{ij}^\ell (2x_j^{\ell-1} - 1) - (2x_i^\ell - 1)) \geq (R_i^\ell - \text{UB}_i^\ell + 1)x_i^\ell, \quad (14a)$$

$$\sum_{j \in J} (W_{ij}^\ell (2x_j^{\ell-1} - 1) - (2x_i^\ell - 1)) \leq (R_i^\ell - \text{LB}_i^\ell - 1)(-x_i^\ell + 1). \quad (14b)$$

Since the number of inequalities described in this theorem grows exponentially with the number of nodes in a hidden layer, these would be used to improve the LP relaxation of the formulation (12) by adding them via a cutting plane procedure.

Lubczyk and Neto (2024) extended Han and Gómez (2021) by investigating the following convex hull for a pair of neurons in the same hidden layer for $\ell \in \{2, \dots, L\}$ and $i, k \in N^\ell$ satisfying $i < k$ in the case where $n^{\ell-1}$ is an even integer larger than 2, every entry in \mathbf{W}^ℓ is nonzero, and $\mathbf{b}^\ell = \mathbf{0}$:

$$\text{conv}(\{(\mathbf{x}^{\ell-1}, x_i^\ell, x_k^\ell) : \mathbf{x}^{\ell-1} \in \{0, 1\}^{n^{\ell-1}}, x_i^\ell = \mathbf{1}_{\mathbb{R}_+}(a_i^\ell(\mathbf{x}^{\ell-1})), x_k^\ell = \mathbf{1}_{\mathbb{R}_+}(a_k^\ell(\mathbf{x}^{\ell-1}))\}).$$

In this work, a relaxation IP problem to the IP problem (11) is solved for a fixed alternative class $t \in N^{L+1}$ to tackle the BNN verification problem. This computational study shows some improvement in the optimal objective value of the LP relaxation of the IP formulation compared to Han and Gómez (2021), but this does not translate to significant improvement in the ability to solve the BNN verification problem.

3 Layerwise Derived Valid Inequalities

We explore an alternative approach for deriving valid inequalities for (11) that is based on recursively approximating the set of “reachable vectors” at each layer.

Starting from X^0 , X^ℓ is defined recursively as follows for $\ell \in [L]$:

$$X^\ell = \mathbf{1}_{\mathbb{R}_+}(a^\ell(X^{\ell-1})).$$

Then, X^ℓ is the set of all output vectors of layer ℓ that can be obtained from a perturbed input vector in the set X^0 . Since the objective (2a) only depends on \mathbf{x}^L , (2) can be formulated as the following IP problem:

$$\begin{aligned} \max_{\mathbf{x}^L} \quad & \max \left\{ a_t^{L+1}(\mathbf{x}^L) - a_{\bar{t}}^{L+1}(\mathbf{x}^L) : t \in N^{L+1} \setminus \{\bar{t}\} \right\} \\ \text{s.t.} \quad & \mathbf{x}^L \in X^L. \end{aligned} \quad (15)$$

The above IP problem implies that (2) can be solved with access to a description for X^L , which motivates studying valid inequalities for this set. We observe that valid inequalities for X^ℓ can be obtained using a relaxation of the set $X^{\ell-1}$ since $X^\ell = \mathbf{1}_{\mathbb{R}_+}(a^\ell(X^{\ell-1}))$. Since a full description for X^0 is given, this motivates an iterative approach in which we derive valid inequalities for the set X^ℓ based on an outer approximation of the set $X^{\ell-1}$ for $\ell = 2, \dots, L$. We refer to such inequalities as *layerwise derived valid inequalities*.

3.1 Variable Fixing

We first investigate layerwise derived valid inequalities for X^ℓ for $\ell \in [L]$ of the form

$$x_i^\ell = \frac{1 - c_i}{2} \quad (16)$$

for $i \in N^\ell$ and $c_i \in \{-1, 1\}$. In the case of $c_i = -1$, (16) is equivalent to $x_i^\ell = 1$ and (16) is equivalent to $x_i^\ell = 0$ in the case of $c_i = 1$. For these reasons, valid equalities for X^ℓ of the form (16) are called variable fixings. This idea of identifying variable fixings is related to the work of Fischetti and Jo (2018) who apply a similar technique to fix binary variables in the IP formulation of the feedforward neural network verification problem.

Our approach is to consider each equality of the form (16) and determine if it is a valid equality. The following theorem describes an IP problem that can be solved to determine validity of such an equality.

Theorem 4. Consider $\ell \in [L]$, $i \in N^\ell$, and $c_i \in \{-1, 1\}$. Let $X_{\text{out}}^{\ell-1} \subset \{0, 1\}^{n^{\ell-1}}$ satisfy $X_{\text{out}}^{\ell-1} \supseteq X^{\ell-1}$ and define

$$z^* = \max \left\{ c_i \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1} : \mathbf{x}^{\ell-1} \in X_{\text{out}}^{\ell-1} \right\}. \quad (17)$$

If $z^* \leq c_i R_i^\ell / 2$, where R_i^ℓ is defined in (3), then (16) is a valid equality for X^ℓ .

The details are described in Section 3.3, but to preview this, the idea is to use Theorem 4 to obtain outer approximations of the sets X^ℓ by setting X_{out}^0 as X^0 , and then recursively using the theorem to derive valid inequalities to define valid inequalities to define X_{out}^ℓ based on the previously determined $X_{\text{out}}^{\ell-1}$ for $\ell = 1, \dots, L$.

To prove this theorem, we use the following lemma.

Lemma 5. Consider $\ell \in [L]$, $i \in N^\ell$, and $c_i \in \{-1, 1\}$. Then $\mathbf{x}^{\ell-1} \in \{0, 1\}^{n^{\ell-1}}$ satisfies $c_i \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1} \leq c_i R_i^\ell / 2$ if and only if $\mathbf{1}_{\mathbb{R}_+}(a_i^\ell(\mathbf{x}^{\ell-1})) = \frac{1 - c_i}{2}$.

. Proof. In the case of $c_i = -1$, the following statements hold for $\ell = 1$:

$$\begin{aligned} & 2c_i \sum_{j \in N^0} W_{ij}^1 x_j^0 \leq c_i R_i^1, \\ \Leftrightarrow & 2 \sum_{j \in N^0} W_{ij}^1 x_j^0 \geq R_i^1, \\ \Leftrightarrow & 2 \sum_{j \in N^0} W_{ij}^1 x_j^0 \geq R_i^1 + \frac{1}{q}, & (2 \sum_{j \in N^0} W_{ij}^1 x_j^0, R_i^1 + \frac{1}{q} \in \frac{2}{q}\mathbb{Z}) \\ \Leftrightarrow & 2 \sum_{j \in N^0} W_{ij}^1 x_j^0 \geq \sum_{j \in N^0} W_{ij}^1 - b_i^1, & (R_i^1 + \frac{1}{q} \text{ is the smallest element of } \frac{2}{q}\mathbb{Z} \\ & \text{not smaller than } \sum_{j \in N^0} W_{ij}^1 - b_i^1) \\ \Leftrightarrow & a_i^1(\mathbf{x}^0) \geq 0, \\ \Leftrightarrow & \mathbf{1}_{\mathbb{R}_+}(a_i^1(\mathbf{x}^0)) = 1 = \frac{1 - c_i}{2}. \end{aligned}$$

For $\ell \in \{2, \dots, L\}$, the same arguments hold because $2 \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1}$ and $R_i^\ell + 1$ are even integers, and $R_i^\ell + 1$ is the smallest even integer not smaller than $\sum_{j \in N^{\ell-1}} W_{ij}^\ell - b_i^\ell$.

In the case of $c_i = 1$, the following statements hold for $\ell = 1$:

$$\begin{aligned}
& 2c_i \sum_{j \in N^0} W_{ij}^1 x_j^0 \leq c_i R_i^1, \\
\Leftrightarrow & 2 \sum_{j \in N^0} W_{ij}^1 x_j^0 \leq R_i^1, \\
\Leftrightarrow & 2 \sum_{j \in N^0} W_{ij}^1 x_j^0 < R_i^1 + \frac{1}{q}, & (2 \sum_{j \in N^0} W_{ij}^1 x_j^0, R_i^1 + \frac{1}{q} \in \frac{2}{q}\mathbb{Z}) \\
\Leftrightarrow & 2 \sum_{j \in N^0} W_{ij}^1 x_j^0 < \sum_{j \in N^0} W_{ij}^1 - b_i^1, & (R_i^1 + \frac{1}{q} \text{ is the smallest element in } \frac{2}{q}\mathbb{Z} \\
& & \text{not smaller than } \sum_{j \in N^0} W_{ij}^1 - b_i^1) \\
\Leftrightarrow & a_i^1(\mathbf{x}^0) < 0, \\
\Leftrightarrow & \mathbf{1}_{\mathbb{R}_+}(a_i^1(\mathbf{x}^0)) = 0 = \frac{1 - c_i}{2}.
\end{aligned}$$

For $\ell \in \{2, \dots, L\}$, the same arguments hold because $2 \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1}$ and $R_i^\ell + 1$ are even integers, and $R_i^\ell + 1$ is the smallest even integer not smaller than $\sum_{j \in N^{\ell-1}} W_{ij}^\ell - b_i^\ell$. \square

. Proof of Theorem 4. Let $\mathbf{x}^\ell \in X^\ell$ and let $\mathbf{x}^{\ell-1} \in X^{\ell-1}$ be such that $\mathbf{x}^\ell = \mathbf{1}_{\mathbb{R}_+}(a^\ell(\mathbf{x}^{\ell-1}))$. Then $\mathbf{x}^{\ell-1} \in X_{\text{out}}^{\ell-1}$ because $X_{\text{out}}^{\ell-1} \supseteq X^{\ell-1}$. Then $z^* \leq c_i R_i/2$ implies

$$c_i \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1} \leq c_i R_i/2$$

which by Lemma 5 implies

$$x_i^\ell = \mathbf{1}_{\mathbb{R}_+}(a_i^\ell(\mathbf{x}^{\ell-1})) = \frac{1 - c_i}{2}.$$

Therefore, (16) is a valid equality for X^ℓ . \square

Although checking whether a variable fixing is valid requires solving the IP (17), this IP is much easier to solve than the original verification IP as it considers only a single layer at a time and the constraints defining the outer approximation of $X^{\ell-1}$ do not involve any “big- M ” constraint (17). Furthermore, it can be solved very efficiently for certain structures of $X_{\text{out}}^{\ell-1}$ as discussed in the following paragraphs.

First, consider the case $\ell \in \{2, \dots, L\}$ and assume $X_{\text{out}}^{\ell-1}$ is defined only by variable fixings, e.g., as is the case if it is defined recursively only the result of Theorem 4. Then, (17) can be solved by setting $x_j^{\ell-1}$ to 1 if $c_i W_{ij}^\ell = 1$ and the variable $x_j^{\ell-1}$ is not fixed in $X_{\text{out}}^{\ell-1}$, and setting all other unfixed variables to 0.

For $\ell = 1$, the special form of the set X^0 also allows (17) to be solved efficiently. We describe here how this is done for the case $p = 1$. The methods for solving (17) for the cases $p = 2$ and $p = \infty$ are described in Appendix A.4. For $p = 1$, the following inequality holds because $\mathbf{x}^0 \in [0, 1]^{n^0}$:

$$c_i \sum_{j \in N^0} W_{ij}^1 x_j^0 \leq \sum_{j \in N^0} \max(c_i W_{ij}^1, 0). \quad (18)$$

Inequality (18) is satisfied at equality by $\hat{\mathbf{x}}$ defined by $\hat{x}_j = \bar{x}_j$ for j with $W_{ij}^1 = 0$, $\hat{x}_j = 1$ for j with $W_{ij}^1 = c_i$, and $\hat{x}_j = 0$ for j with $W_{ij}^1 = -c_i$. If $\|\hat{\mathbf{x}} - \bar{\mathbf{x}}\|_1 \leq \epsilon$, $\hat{\mathbf{x}} \in X^0$, so the optimal objective

value is $\sum_{j \in N^0} \max(c_i W_{ij}^1, 0)$. Otherwise, by iteratively decreasing \hat{x}_j by $\frac{1}{q}$ for j with $W_{ij}^1 = c_i$ and $\hat{x}_j > \bar{x}_j$ or increasing \hat{x}_j by $\frac{1}{q}$ for j with $W_{ij}^1 = -c_i$ and $\hat{x}_j < \bar{x}_j$ we can obtain a solution $\mathbf{x}^* \in X^0$ satisfying $\|\mathbf{x}^* - \bar{\mathbf{x}}\|_1 = \frac{1}{q} \lfloor q\epsilon \rfloor$, $x_j^* = \bar{x}_j$ for j with $W_{ij}^1 = 0$, $x_j^* > \bar{x}_j$ for j with $W_{ij}^1 = c_i$, and $x_j^* < \bar{x}_j$ for j with $W_{ij}^1 = -c_i$. The following inequalities are satisfied by every $\mathbf{x} \in X^0$ and hold at equality for \mathbf{x}^* :

$$\begin{aligned} c_i \sum_{j \in N^0} W_{ij}^1 x_j^0 &= \sum_{j \in N^0} c_i W_{ij}^1 (x_j^0 - \bar{x}_j) + c_i \sum_{j \in N^0} W_{ij}^1 \bar{x}_j \\ &\leq \sum_{j \in N^0} |x_j^0 - \bar{x}_j| + c_i \sum_{j \in N^0} W_{ij}^1 \bar{x}_j && (c_i W_{ij}^1 \in \{-1, 0, 1\}) \\ &\leq \frac{1}{q} \lfloor q\epsilon \rfloor + c_i \sum_{j \in N^0} W_{ij}^1 \bar{x}_j \end{aligned} \quad (19)$$

where the last inequality follows because $\sum_{j \in N^0} |x_j^0 - \bar{x}_j| = \|\mathbf{x}^0 - \bar{\mathbf{x}}\|_1 \leq \epsilon$ and $\sum_{j \in N^0} |x_j^0 - \bar{x}_j|$ is a multiple of $\frac{1}{q}$. It follows that the optimal objective value of (17) in this case is $\frac{1}{q} \lfloor q\epsilon \rfloor + c_i \sum_{j \in N^0} W_{ij}^1 \bar{x}_j$ and this value is achieved by \mathbf{x}^* .

3.2 Two-variable Inequalities

We next seek to derive sufficient conditions for inequalities of the following form to be valid inequalities for X^ℓ for $\ell \in [L]$:

$$c_i x_i^\ell + c_k x_k^\ell \leq \frac{c_i + c_k}{2}, \quad (20)$$

for $i, k \in N^\ell$ satisfying $i > k$, and $c_i, c_k \in \{-1, 1\}$. The following theorem presents an IP that yields a sufficient condition for (20) to be a valid inequality for X^ℓ .

Theorem 6. Consider $\ell \in [L]$, $i, k \in N^\ell$ satisfying $i < k$, and $c_i, c_k \in \{-1, 1\}$. Let $X_{\text{out}}^{\ell-1} \subset \{0, 1\}^{n^{\ell-1}}$ satisfy $X_{\text{out}}^{\ell-1} \supseteq X^{\ell-1}$ and define

$$z^* = \max \left\{ c_i \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1} : \mathbf{x}^{\ell-1} \in X_{\text{out}}^{\ell-1}, c_k \sum_{j \in N^{\ell-1}} W_{kj}^\ell x_j^{\ell-1} \geq c_k R_k^\ell / 2 \right\}. \quad (21)$$

If $z^* \leq c_i R_i^\ell / 2$, where R_i^ℓ is defined in (3), then (20) is a valid inequality for X^ℓ .

Proof. Let $\mathbf{x}^\ell \in X^\ell$ and let $\mathbf{x}^{\ell-1} \in X^{\ell-1}$ be such that $\mathbf{x}^\ell = \mathbf{1}_{\mathbb{R}_+}(a^\ell(\mathbf{x}^{\ell-1}))$ and observe $\mathbf{x}^{\ell-1} \in X_{\text{out}}^{\ell-1}$ since $X_{\text{out}}^{\ell-1} \supseteq X^{\ell-1}$.

Observe that for a binary variable x and $c \in \{-1, 1\}$, it holds that $cx \in \{\frac{c+1}{2}, \frac{c-1}{2}\}$.

Suppose that $c_k x_k^\ell = \frac{c_k - 1}{2}$. Then,

$$c_i x_i^\ell + c_k x_k^\ell \leq \frac{c_i + 1}{2} + \frac{c_k - 1}{2} = \frac{c_i + c_k}{2}$$

and hence (20) is satisfied.

Thus, assume now that $c_k x_k^\ell = \frac{c_k + 1}{2}$, and hence $x_k^\ell = \frac{c_k + 1}{2}$ because $c_k \in \{-1, 1\}$. Then Lemma 5 implies

$$c_k \sum_{j \in N^{\ell-1}} W_{kj}^\ell x_j^{\ell-1} > c_k R_k^\ell / 2$$

and hence $\mathbf{x}^{\ell-1}$ is feasible to (21). Hence,

$$c_i \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1} \leq z^* \leq c_i R_i^\ell / 2$$

which again by Lemma 5 implies

$$x_i^\ell = \mathbf{1}_{\mathbb{R}_+}(a_i^\ell(\mathbf{x}^{\ell-1})) = \frac{1 - c_i}{2}.$$

Thus, $c_i x_i^\ell = \frac{c_i - 1}{2}$ and hence

$$c_i x_i^\ell + c_k x_k^\ell = \frac{c_i - 1}{2} + \frac{c_k + 1}{2} = \frac{c_i + c_k}{2}$$

and hence gain (20) is satisfied. \square

Once again, solving (20) is expected to be much simpler than solving the original verification problem as it only includes the constraints defining the outer approximation of $X^{\ell-1}$ and one other constraint and does not contain any “big- M ” constraints.

3.3 Algorithms

We now describe how we propose to use the layerwise derived valid inequalities to solve the IP problem for BNN verification.

Algorithm 1: VerifyBnn($\bar{\mathbf{x}}, \epsilon$)

Input: input vector $\bar{\mathbf{x}}$, input perturbation ϵ
Output: if $\bar{\mathbf{x}}$ is ϵ -verified, returns TRUE, else returns FALSE

- 1 Initialize $X_{\text{in}} = (X_{\text{in}}^0, \dots, X_{\text{in}}^L) \leftarrow (\emptyset, \dots, \emptyset)$
- 2 $X_{\text{in}} \leftarrow \text{UpdateInApprox}(0, X_{\text{in}}, \{\bar{\mathbf{x}}\})$
- 3 Initialize $X_{\text{out}}^0 \leftarrow X^0$
- 4 **for** $\ell = 1, \dots, L$ **do**
- 5 $X_{\text{in}}, N_{\text{fix}}^\ell, X_{\text{out}}^\ell \leftarrow \text{PhaseOneFixVar}(\ell, X_{\text{in}}, X_{\text{out}}^{\ell-1})$
- 6 Add constraints $\mathbf{x}^\ell \in X_{\text{out}}^\ell$ for $\ell \in [L]$ to (12) and start the solution process but with a limit of one node, and let z^* be the best objective value (lower bound) and \bar{z} be the best upper bound on the optimal objective value
- 7 **if** $\bar{z} \leq 0$ **then**
- 8 **return** TRUE
- 9 **else if** $z^* > 0$ **then**
- 10 **return** FALSE
- 11 **else**
- 12 $X_{\text{in}}, X_{\text{out}}^1 \leftarrow \text{GenTwoVarIneq}(1, X_{\text{in}}, X_{\text{out}}^0, N_{\text{fix}}^1, X_{\text{out}}^1)$
- 13 **for** $\ell = 2, \dots, L$ **do**
- 14 $X_{\text{in}}, N_{\text{fix}}^\ell, X_{\text{out}}^\ell \leftarrow \text{PhaseTwoFixVar}(\ell, X_{\text{in}}, X_{\text{out}}^{\ell-1}, N_{\text{fix}}^\ell, X_{\text{out}}^\ell)$
- 15 $X_{\text{in}}, X_{\text{out}}^\ell \leftarrow \text{GenTwoVarIneq}(\ell, X_{\text{in}}, X_{\text{out}}^{\ell-1}, N_{\text{fix}}^\ell, X_{\text{out}}^\ell)$
- 16 Solve (12) by adding constraints $\mathbf{x}^\ell \in X_{\text{out}}^\ell$ for $\ell \in [L]$, and obtain $z_\epsilon^*(\bar{\mathbf{x}})$ from the optimal objective value
- 17 **if** $z_\epsilon^*(\bar{\mathbf{x}}) \leq 0$ **then**
- 18 **return** TRUE
- 19 **else**
- 20 **return** FALSE

Our proposed method is described in Algorithm 1. The algorithm consists of two phases. In the first phase, we only try to fix variables, since in this case the validity verification problem (17) is easy to solve. This is described in lines 3 - 5, where variable fixings are identified by calling Algorithm 2 for each $\ell \in [L]$, which outputs an outer approximation for X^ℓ (X_{out}^ℓ) based on fixing variables in the set $N_{\text{fix}}^\ell \subseteq N^\ell$. Then, the generated variable fixings are added to the IP formulation (12) as constraints and (12) is solved by an IP solver with a limit of one node (i.e., the IP solver only processes the root node of the branch-and-bound tree). If the IP solver solves the verification problem within that limit (e.g., by finding a feasible solution with positive objective value or proving a non-positive upper bound), then it returns the result accordingly. Otherwise, Algorithm 1 moves to the second phase. In this phase, two-variable inequalities are generated by Algorithm 4 for $\ell \in [L]$ and additional variable fixings are checked, which may be possible due to the improved outer approximations defined by the two-variable inequalities. For layers $\ell \in \{2, \dots, L\}$, Algorithm 4 outputs an outer approximation X_{out}^ℓ that is potentially a subset of the one generated in the first phase. The inequalities defining these outer approximations are added to the IP formulation (12) and it is solved. In order to answer the verification question, it is only necessary to determine the sign of the optimal value of (12). Thus, when solving (12) the process can be terminated if either the best objective value of a feasible solution found is positive or the best upper bound for the optimal objective value is non-positive.

Algorithm 2: PhaseOneFixVar($\ell, X_{\text{in}}, X_{\text{out}}^{\ell-1}$)

Input: layer ℓ , inner approximation X_{in} , outer approximation $X_{\text{out}}^{\ell-1}$ for $X^{\ell-1}$

Output: X_{in} , set N_{fix}^ℓ of $i \in N^\ell$ where x_i^ℓ is fixed in X^ℓ , outer approximation X_{out}^ℓ for X^ℓ

1 Initialize $N_{\text{fix}}^\ell \leftarrow \emptyset$

2 Initialize $X_{\text{out}}^\ell \leftarrow \{0, 1\}^{n^\ell}$

3 **for** $i \in N^\ell$ **do**

4 **for** $c_i \in \{-1, 1\}$ **do**

5 Solve the IP subproblem (17), and obtain the optimal objective value z^*

6 **if** $\ell = 1$ **then**

7 Let x^{i, c_i} be optimal solution obtained when solving (17)

8 $X_{\text{in}} \leftarrow \text{UpdateInApprox}(0, X_{\text{in}}, \{x^{i, c_i}\})$

9 **if** $z^* \leq c_i R_i^\ell / 2$ **then**

10 $N_{\text{fix}}^\ell \leftarrow N_{\text{fix}}^\ell \cup \left\{ \left(i, \frac{-c_i + 1}{2} \right) \right\}$

11 $X_{\text{out}}^\ell \leftarrow X_{\text{out}}^\ell \cap \left\{ \mathbf{x}^\ell \in \{0, 1\}^{n^\ell} : x_i^\ell = \frac{-c_i + 1}{2} \right\}$

12 **return** $X_{\text{in}}, N_{\text{fix}}^\ell, X_{\text{out}}^\ell$

Algorithms 2, 3, and 4 generate variable fixings in phase 1, variable fixings in phase 2, and two-variable inequalities, respectively. These algorithms check whether each candidate layerwise derived valid inequality is valid for X^ℓ by solving the IP subproblem (17) or (20). When solving these IP subproblems, if the upper bound on the optimal objective value falls below $c_i R_i^\ell / 2$ the optimization process is terminated since we then know the candidate inequality is valid. On the other hand, if the lower bound (incumbent objective value) exceeds $c_i R_i^\ell / 2$, the optimization process is terminated since we can then conclude that we are not able to verify validity of the candidate inequality. In Algorithms 2 and 3, the neuron fixed by the candidate is added to N_{fix}^ℓ because this set is exploited in Algorithms 3 and 4 to retrieve candidates for layerwise derived valid inequalities.

Algorithm 3: PhaseTwoFixVar($\ell, X_{\text{in}}, X_{\text{out}}^{\ell-1}, N_{\text{fix}}^{\ell}, X_{\text{out}}^{\ell}$)

Input: layer ℓ , inner approximation X_{in} , outer approximation $X_{\text{out}}^{\ell-1}$ for $X^{\ell-1}$, set N_{fix}^{ℓ} of $i \in N^{\ell}$ where x_i^{ℓ} is fixed in X^{ℓ} , outer approximation X_{out}^{ℓ} for X^{ℓ}

Output: $X_{\text{in}}, N_{\text{fix}}^{\ell}, X_{\text{out}}^{\ell}$

```
1 for  $i \in N^{\ell} \setminus N_{\text{fix}}^{\ell}$  do
2   for  $c_i \in \{-1, 1\}$  do
3     if  $\max_{\mathbf{x}^{\ell} \in X_{\text{in}}^{\ell}} c_i x_i^{\ell} > \frac{c_i - 1}{2}$  then
4       continue
5     Solve the IP subproblem (17), and obtain a set  $X_{\text{feas}}^{\ell-1}$  of feasible solutions and the
      optimal objective value  $z^*$ 
6      $X_{\text{in}} \leftarrow \text{UpdateInApprox}(\ell, X_{\text{in}}, \mathbf{1}_{\mathbb{R}_+}(a^{\ell}(X_{\text{feas}}^{\ell-1})))$ 
7     if  $z^* \leq c_i R_i^{\ell}/2$  then
8        $N_{\text{fix}}^{\ell} \leftarrow N_{\text{fix}}^{\ell} \cup \{(i, \frac{-c_i + 1}{2})\}$ 
9        $X_{\text{out}}^{\ell} \leftarrow X_{\text{out}}^{\ell} \cap \{\mathbf{x}^{\ell} \in \{0, 1\}^{n^{\ell}} : x_i^{\ell} = \frac{-c_i + 1}{2}\}$ 
10 return  $X_{\text{in}}, N_{\text{fix}}^{\ell}, X_{\text{out}}^{\ell}$ 
```

Algorithm 4: GenTwoVarIneq($\ell, X_{\text{in}}, X_{\text{out}}^{\ell-1}, N_{\text{fix}}^{\ell}, X_{\text{out}}^{\ell}$)

Input: layer ℓ , inner approximation X_{in} , outer approximation $X_{\text{out}}^{\ell-1}$ for $X^{\ell-1}$, set N_{fix}^{ℓ} of $i \in N^{\ell}$ where x_i^{ℓ} is fixed in X^{ℓ} , outer approximation X_{out}^{ℓ} for X^{ℓ}

Output: $X_{\text{in}}, X_{\text{out}}^{\ell}$

```
1 Initialize  $N_{\text{pair}}^{\ell} \leftarrow \{(i, c_i, k, c_k) : i, k \in N^{\ell} \setminus N_{\text{fix}}^{\ell}, c_i, c_k \in \{-1, 1\}, i < k\}$ , fail_count  $\leftarrow 0$ 
2 Sort  $N_{\text{pair}}^{\ell}$  in a descending order by the score based on  $X_{\text{in}}^{\ell}$ 
3 for  $(i, c_i, k, c_k) \in N_{\text{pair}}^{\ell}$  in descending order of score( $i, c_i, k, c_k$ ) do
4   if  $\max_{\mathbf{x}^{\ell} \in X_{\text{in}}^{\ell}} (c_i x_i^{\ell} + c_k x_k^{\ell}) > \frac{c_i + c_k}{2}$  then
5     continue
6   Solve the IP subproblem (21), and obtain a set  $X_{\text{feas}}^{\ell-1}$  of feasible solutions and the
      optimal objective value  $z^*$ 
7    $X_{\text{in}} \leftarrow \text{UpdateInApprox}(\ell, X_{\text{in}}, \mathbf{1}_{\mathbb{R}_+}(a^{\ell}(X_{\text{feas}}^{\ell-1})))$ 
8   if  $z^* \leq c_i R_i^{\ell}/2$  then
9      $X_{\text{out}}^{\ell} \leftarrow X_{\text{out}}^{\ell} \cap \{\mathbf{x}^{\ell} \in \{0, 1\}^{n^{\ell}} : c_i x_i^{\ell} + c_k x_k^{\ell} \leq \frac{c_i + c_k}{2}\}$ 
10    fail_count  $\leftarrow 0$ 
11  else
12    fail_count  $\leftarrow$  fail_count + 1
13  if fail_count  $\geq$  max_fail then
14    break
15 return  $X_{\text{in}}, X_{\text{out}}^{\ell}$ 
```

Algorithm 5: UpdateInApprox($\ell, X_{\text{in}}, X_{\text{feas}}^\ell$)

Input: layer ℓ , inner approximation X_{in} , set $X_{\text{feas}}^\ell \subseteq X_{\text{out}}^\ell$
Output: X_{in}
1 $X_{\text{in}}^\ell \leftarrow X_{\text{in}}^\ell \cup X_{\text{feas}}^\ell$
2 **for** $\ell' = \ell + 1, \dots, L$ **do**
3 Initialize $X_{\text{feas}}^{\ell'} \leftarrow \mathbf{1}_{\mathbb{R}_+}(a^{\ell'}(X_{\text{feas}}^{\ell'-1}))$
4 $X_{\text{in}}^{\ell'} \leftarrow X_{\text{in}}^{\ell'} \cup X_{\text{feas}}^{\ell'}$
5 **return** X_{in}

To limit the time spent checking validity of candidates, Algorithms 3 and 4 use an inner approximation X_{in}^ℓ for $\mathbf{1}_{\mathbb{R}_+}(a^\ell(X_{\text{out}}^{\ell-1}))$ to detect candidates that will not lead to a valid inequality, and hence avoid solving (17) or (20) for those candidates. For each candidate layerwise derived inequality, if the maximum of the left-hand side over X_{in}^ℓ is larger than the right-hand side, this candidate is not valid for X^ℓ , so solving the IP subproblem can be avoided. In Algorithm 1, $X_{\text{in}} = (X_{\text{in}}^0, \dots, X_{\text{in}}^L)$ is initialized by propagating the input feature vector $\bar{\mathbf{x}}$ through the BNN. In Algorithm 2 these sets are expanded by propagating the optimal solution obtained when solving (17) at layer 1 (which are in X^0 by definition) through the BNN, thus yielding solutions in X^ℓ which are added to X_{in}^ℓ for $\ell = 0, \dots, L$. In Algorithms 3 and 4, for each layer $\ell \in \{1, \dots, L\}$, each time we solve (17) or (20) to check validity of a candidate inequality we collect the set of feasible solutions $X_{\text{feas}}^{\ell-1}$ obtained by the solver, which by definition are in the set $X_{\text{out}}^{\ell-1}$. These are then propagated forward as $X_{\text{feas}}^{\ell'} = \mathbf{1}_{\mathbb{R}_+}(a^{\ell'}(X_{\text{feas}}^{\ell'-1}))$ for $\ell' = \ell, \dots, L$ and these sets are added to $X_{\text{in}}^{\ell'}$. Since the set $X_{\text{out}}^{\ell-1}$ is completely determined when processing layer ℓ in Algorithm 3, this process ensures that $X_{\text{in}}^\ell \subseteq X_{\text{out}}^\ell$ for all ℓ , and hence if a candidate valid inequality is violated by a vector in X_{in}^ℓ we can conclude that solving (17) or (20) cannot lead to a verification that the inequality is valid.

As a final strategy to limit the computational time spent solving (20) on candidates that do not yield valid inequalities, Algorithm 4 includes a heuristic stopping rule that quits the process if it seems unlikely to yield more additional valid inequalities. The idea is to check the candidate inequalities in a sequence defined by a score that correlates with how likely they are to yield a valid inequality, and then terminate once the number of consecutive failed attempts exceeds a pre-specified limit ‘max_fail’. The score that we use for a candidate inequality $(i, c_i, k, c_k) \in N_{\text{pair}}^\ell$ also leverages the inner approximations we have built and is defined as:

$$\text{score}(i, c_i, k, c_k) = \frac{|\{\hat{\mathbf{x}}^\ell \in X_{\text{in}}^\ell : \hat{x}_i^\ell = \frac{-c_i+1}{2}\}| + |\{\hat{\mathbf{x}}^\ell \in X_{\text{in}}^\ell : \hat{x}_k^\ell = \frac{-c_k+1}{2}\}|}{|X_{\text{in}}^\ell|}.$$

To understand the intuition for this score, consider the case of $c_i = c_k = 1$, so that the candidate inequality is of the form $x_i^\ell + x_k^\ell \leq 1$, and the score sums the fraction of solutions in the inner approximation X_{in}^ℓ which have $x_i^\ell = 0$ and which have $x_k^\ell = 0$. A high score suggests most solutions in X_{in}^ℓ have either $x_i^\ell = 0$ or $x_k^\ell = 0$ and hence the candidate inequality might be satisfied. The intuition behind the other cases is similar.

4 Computational Study

We pursue a computational study to investigate how IP methods work to solve the BNN verification problem for multiple test instances. The following five IP methods are considered in the

Network	L	n ¹	n ²	n ³	n ⁴	Error Rate
1	2	100	100	*	*	7.53%
2	2	200	100	*	*	5.52%
3	2	300	200	*	*	4.19%
4	3	100	100	100	*	7.33%
5	3	200	100	100	*	5.08%
6	3	300	200	100	*	3.68%
7	4	200	100	100	100	5.04%
8	4	300	200	100	100	3.46%
9	4	500	300	200	100	2.32%

Table 1: Networks in computational study

computational study:

- Many-IP: solve the IP problem (11) for all t ,
- 1-IP: solve the IP problem (12),
- 1-IP+HG: solve (12) by employing a constraint generation approach with (14),
- 1-IP+Fix: solve (12) by employing the variant of Algorithm 1 where non-root nodes are explored in solving (12) in phase 1, and phase 2 is skipped,
- 1-IP+Fix+2Var: solve (12) by employing Algorithm 1.

Many-IP and 1-IP are compared to explore the impact of the technique for obtaining a linear objective. The methods 1-IP, 1-IP+HG, 1-IP+Fix, and 1-IP+Fix+2Var are compared to explore the impact of the techniques for generating layerwise derived valid inequalities.

4.1 Test Instances

Unless stated otherwise, the maximum perturbation from $\bar{\mathbf{x}}$ is defined using an ℓ_1 norm in our test instances.

Each test instance in the computational study consists of a BNN, $\bar{\mathbf{x}}$, and ϵ . Nine BNNs are trained for the computational study by using Hubara et al. (2016)’s method based on a gradient descent method with the MNIST train dataset. The MNIST dataset consists of feature vectors representing handwritten digits from 0 to 9, so $n^{L+1} = 10$. These feature vectors are originally 784-dimensional non-negative integer vectors whose coordinates are at most 255, but they are scaled to 784-dimensional non-negative real vectors whose coordinates are at most 1, so $n^0 = 784$ and $q = 255$. The number of hidden layers, the number of neurons in hidden layers, and the error rate of each network are reported in Table 1. The error rate of each network is computed as the portion of feature vectors misclassified by this network in the MNIST test dataset.

For each network, we consider 10 instances, defined by 10 different feature vectors $\bar{\mathbf{x}}$. For each digit from 0 to 9, one feature vector in the MNIST test dataset whose \bar{t} is this digit is randomly chosen.

For each network and feature vector $\bar{\mathbf{x}}$, six values are obtained for ϵ by employing Algorithm 6 with $\text{max_iter} = 6$. Algorithm 6 is a binary search that attempts to find the largest ϵ under which $\bar{\mathbf{x}}$ can be ϵ -verified using a given method and time limit in each iteration. In Algorithm 6, if $\bar{\mathbf{x}}$ is ϵ -verified for the current ϵ , ϵ is increased based on the smallest input perturbation ϵ_{UB}

Algorithm 6: Perturbation Selection

Input: $\epsilon_{\text{init}}, \text{max_iter}$
Output: $\epsilon_{\text{LB}}, \epsilon_{\text{UB}}$: lower and upper bound on maximum value of ϵ for which \bar{x} is ϵ -verified

```
1  $\epsilon_{\text{LB}} \leftarrow 0, \epsilon_{\text{UB}} \leftarrow \text{NULL}$ 
2  $\epsilon \leftarrow \epsilon_{\text{init}}$ 
3 for  $i = 1, \dots, \text{max\_iter}$  do
4   Attempt to solve (1) with a time limit for  $\bar{x}$  and  $\epsilon$  and let  $\bar{z}$  be the best upper bound
   obtained.
5   if  $\bar{z} \leq 0$  then
6      $\epsilon_{\text{LB}} \leftarrow \epsilon$ 
7     if  $\epsilon_{\text{UB}}$  is NULL then
8        $\epsilon \leftarrow 2\epsilon$ 
9     else
10       $\epsilon \leftarrow \frac{\epsilon + \epsilon_{\text{UB}}}{2}$ 
11   else
12      $\epsilon_{\text{UB}} \leftarrow \epsilon$ 
13      $\epsilon \leftarrow \frac{\epsilon + \epsilon_{\text{LB}}}{2}$ 
14 return  $\epsilon, \epsilon_{\text{LB}}, \epsilon_{\text{UB}}$ 
```

under which \bar{x} may not be ϵ_{UB} -verified. Otherwise, ϵ is decreased to the average of ϵ and the largest input perturbation ϵ_{LB} under which \bar{x} is ϵ_{LB} -verified. Different IP methods for the BNN verification may result in different sequences of ϵ in Algorithm 6 based on their ability to solve the verification problem at a given ϵ within the time limit. This approach is used for determining the ϵ values in the test instances in order to find challenging instances, i.e., those in which determining whether \bar{x} is ϵ -verified is nontrivial. The initial value ϵ_{init} is set to 1 in the case that the maximum perturbation from \bar{x} is defined by an ℓ_1 norm, $\frac{1}{255}$ in the case that the maximum perturbation is defined by ℓ_∞ norm, and $\frac{1}{32}$ in the case that the maximum perturbation is defined by ℓ_2 norm.

4.2 Implementation Details

In the IP method 1-IP+HG, (14) are generated by solving the LP relaxation problem of the IP problem (12) and adding violated inequalities (14a) and (14b) in an iterative manner. In each iteration, for each $\ell \in \{2, \dots, L\}$ and $i \in N^\ell$, an inequality (14b) with the largest violation is added to (12). Likewise, (14a) with the largest violation is added to (12) as a constraint. This iteration is repeated until either no violated inequalities are found or the optimal objective value of the LP relaxation problem does not improve by 1% over the last 10 iterations. Our test instances are different than those used in Han and Gómez (2021) (ours have more layers in the BNN and non-binary inputs) and we obtain qualitatively different results than those presented in Han and Gómez (2021). Thus, to validate our implementation, in Appendix A.6 we present results of additional experiments with the method 1-IP+HG on the instances used in Han and Gómez (2021) which indicates that our implementation achieves qualitatively similar results to what is reported in Han and Gómez (2021) on those instances.

In the IP method 1-IP+Fix and 1-IP+Fix+2Var, in Algorithm 2, the IP subproblem (17) is solved using the method described at the end of Section 3.1 (i.e., without an IP solver).

In 1-IP+Fix+2Var, two-variable inequalities for the first hidden layer are not generated, and variable fixings for the second hidden layer are not generated in phase 2 because two-variable

Network	Many-IP	1-IP	1-IP +HG	1-IP +Fix	1-IP +Fix +2Var
1	7.23	7.23	7.23	7.23	7.23
2	6.15	6.48	6.48	6.48	6.48
3	1.60	5.45	5.32	6.28	6.28
4	6.07	6.28	6.28	6.18	6.28
5	5.55	6.35	6.35	6.55	7.35
6	2.95	4.30	3.95	4.43	5.45
7	4.51	5.06	5.06	5.29	6.49
8	2.51	3.89	3.78	3.99	5.28
9	0.64	2.17	2.06	2.34	3.88

Table 2: Average maximum ϵ under which $\bar{\mathbf{x}}$ is ϵ -verified by each method within the time limit.

inequalities for the first hidden layer are unlikely to be generated for the test instances. The networks are dense, and ϵ is much smaller than $n^0 = 784$ in test instances, so for $i, k \in N^1$ satisfying $i < k$ and $c_i, c_k \in \{-1, 1\}$, it is likely to find $\hat{\mathbf{x}}^0 \in X^0$ where $2c_i \sum_{j \in N^0} W_{ij}^1 \hat{x}_j^0$ and $2c_k \sum_{j \in N^0} W_{kj}^1 \hat{x}_j^0$ achieve their maximum over X^0 . If both x_i^1 and x_k^1 are not fixed, $\hat{\mathbf{x}}^0$ violates (20), so two-variable inequalities for the first hidden layer are rarely generated. As a result of not generating two-variable inequalities for the first hidden layer, variable fixings for the second hidden layer cannot be generated in phase 2. We use `max_fail = 100` as the number of consecutive failures after which we terminate attempting to generate two-variable inequalities.

The time limit for all IP methods is 3600 seconds. In the IP method Many-IP, the time limit to solve the IP problem (11) for each t is set to 400 seconds because there are nine alternative classes. In 1-IP+HG, 1-IP+Fix, and 1-IP+Fix+2Var, a time limit on the phase for generating valid inequalities of 2700 seconds is imposed. The time limit to solve (12) after generating the inequalities is set to 3600 seconds *subtract* the time spent generating valid inequalities. If the time limit is hit, the best upper bound for the optimal objective value is used instead of the best objective value to answer the BNN verification problem because $\bar{\mathbf{x}}$ is ϵ -verified if and only if the best upper bound is non-positive.

All IP methods are implemented in Python, and Gurobi 10.0.3 is used as the IP solver. All experiments in the computational study are run on an Ubuntu desktop with 32 GB RAM and 16 Intel Core i7-10700 CPUs running at 2.90 GHz.

4.3 Results

We first collect the maximum ϵ under which $\bar{\mathbf{x}}$ is ϵ -verified for each IP method, network, and $\bar{\mathbf{x}}$ in the case that the maximal perturbation from $\bar{\mathbf{x}}$ is defined using an ℓ_1 norm. The maximum ϵ is obtained from using Algorithm 6 with `max_iter = 6`. For the same test instance, one IP method may succeed in verifying the BNN, but another IP method fails because it may not be able to verify the BNN within the time limit. As a result, the set of ϵ defining the six instances may differ by IP methods for the same network and $\bar{\mathbf{x}}$, and the maximum ϵ may differ.

For each IP method and network, the arithmetic mean of the maximum ϵ over ten $\bar{\mathbf{x}}$ is presented in Table 2. Many-IP achieves the smallest maximum ϵ , and 1-IP+HG achieves a smaller maximum ϵ than the 1-IP. Except for Network 4, 1-IP+Fix results in a larger maximum ϵ than 1-IP. Method 1-IP+Fix+2Var yields the largest maximum ϵ in every case. These results indicate that the using the proposed variable fixing approach yields modest improvement in the ability to find the maximum

Network	# of Instances	Many-IP	1-IP
1	60	153.1(3)	29.4
2	57	721.9(12)	110.5
3	20	1420.8(10)	358.9
4	59	269.7(8)	58.4(2)
5	56	986.5(18)	286.5(8)
6	39	1941.8(18)	568.3(7)
7	56	972.3(22)	279.7(12)
8	34	1654.4(17)	503.1(7)
9	18	2852.2(13)	710.8(3)

Table 3: Verification time (seconds) of Many-IP and 1-IP

Network	# of Instances	1-IP	1-IP +HG	1-IP +Fix	1-IP +Fix +2Var
1	60	29.4	31.6	5.6	7.0
2	60	119.6	124.3	16.8	16.2
3	52	779.5(10)	842.5(13)	58.4(4)	62.7(4)
4	59	58.4(2)	75.4(1)	15.2(2)	11.3
5	55	296.1(9)	377.9(9)	61.7(7)	30.1(1)
6	46	612.2(13)	942.1(14)	130.4(12)	56.3(2)
7	52	267.8(12)	376.7(15)	84.3(12)	36.1(4)
8	36	582.5(12)	722.1(12)	119.9(11)	36.0(2)
9	25	1183.8(11)	1445.2(11)	276.1(9)	68.3(1)

Table 4: Verification time (seconds) of IP methods based on 1-IP

ϵ at which an input can be verified, and using two-variable layer-wise derived inequalities yields significantly more improvement.

We next investigate in more detail the ability of the different methods to solve verification problems for various values of ϵ . We first compare Many-IP and 1-IP. To do so, for each network and feature vector we consider the subset of ϵ values that are solved by both of these methods, and compute the shifted geometric mean of solution times to solve the IP problem (12) (for 1-IP) or to solve (11) for all t (for Many-IP), with a shift of 1. These results are presented in Table 3, where for Many-IP, the number in parenthesis is the number of test instances where the time limit is hit in solving (11) for at least one t and for 1-IP, the number in parenthesis is the number of test instances where the time limit is hit in solving (12). We find that 1-IP solves the verification problem much more quickly than Many-IP, indicating that our technique for obtaining a linear objective leads to faster BNN verification. Considering the superiority of 1-IP over Many-IP, we exclude Many-IP from further comparisons.

In Table 4 we report the geometric average verification times over the instances tested by all IP methods based on 1-IP. The set of test instances in this case is based on the set of ϵ values that were solved by all 1-IP methods. This set of test instances may differ from the set used in the Many-IP and 1-IP comparison due to the exclusion of Many-IP from this comparison. The number in parenthesis is the number of test instances where the time limit is hit when solving (12). We find that 1-IP+HG’s verification times are larger than 1-IP’s, and 1-IP+Fix’s are shorter than

Network	1-IP	1-IP +HG	1-IP +Fix	1-IP +Fix +2Var
1	116.4 %	116.4 %	32.5 %	18.2 %
2	120.8 %	120.8 %	36.0 %	19.9 %
3	131.2 %	132.9 %	50.3 %	30.5 %
4	125.4 %	124.9 %	49.1 %	13.9 %
5	137.4 %	137.8 %	67.4 %	20.5 %
6	150.7 %	153.3 %	82.4 %	25.6 %
7	144.1 %	150.2 %	82.0 %	31.7 %
8	178.8 %	179.3 %	93.8 %	26.9 %
9	180.2 %	180.2 %	109.2 %	13.8 %

Table 5: LP gap of IP methods based on 1-IP

1-IP’s. Thus, we conclude that using inequalities (14) does not accelerate BNN verification, but variable fixings do. We also find that 1-IP+Fix+2Var’s verification times are similar to 1-IP+Fix’s for Networks 1-4, and 1-IP+Fix+2Var’s verification times are shorter than 1-IP+Fix’s for Networks 5-9. This suggests that the two-variable layerwise derived inequalities are more helpful for BNN verification of networks with more hidden layers.

We next investigate the LP gaps of (12) over instances tested by all IP methods based on 1-IP. The LP gap is defined as $(z_{\text{LP}}^* - z^*)/\bar{z}_{\text{UB}}$ where z_{LP}^* is the optimal objective value of the LP relaxation problem of (12) after adding any valid inequalities introduced by the method, z^* is the best objective value from a feasible solution to (12) obtained by the method, and

$$\bar{z}_{\text{UB}} = \max_{\substack{\mathbf{x}^L \in \{0,1\}^{n^L} \\ t \in N^{L+1}}} (a_t^{L+1}(\mathbf{x}^L) - a_t^{L+1}(\mathbf{x}^L)).$$

We use \bar{z}_{UB} as the denominator instead of $|z^*|$ because z^* may be near zero for some instances, which would skew interpretation of these relative optimality gaps when averaged over different instances. The quantity \bar{z}_{UB} is positive by definition and is a natural upper bound on the optimal value of (12). Table 5 reports the average LP gaps for each method. We find that the LP gaps of 1-IP+HG and 1-IP are similar, 1-IP+Fix yields significant improvement over 1-IP, and 1-IP-Fix+2Var yields further significant improvement beyond that. Although 1-IP+HG does not improve LP gaps on these instances, we remark that in the supplemental experiments reported in Appendix A.6 we found that on the instances used in Han and Gómez (2021) the use of the single-neuron valid inequalities did yield modest reduction in the LP gap, although even on these instances this reduction still did not translate into reduced time to solve the verification problem.

Table 6 presents additional results for the IP methods based on 1-IP. These results are averaged over all 445 instances solved by all these methods. The row ‘Preprocessing time’ presents the shifted geometric mean of times to generate layerwise derived valid inequalities (resp. (14)) for 1-IP+Fix and 1-IP+Fix+2Var (resp. 1-IP+HG). The row ‘verification time’ is the shifted geometric mean of times to solve the verification problem – if a method does not solve an instance within the time limit, then the time limit is used for that instance. The row ‘# of solved instances (veri.)’ reports the number of instances for which each IP method solved the verification problem within the time limit. Row ‘# of nodes (veri.)’ is the shifted geometric mean of the number of nodes in the branch-and-bound tree when solving the verification problem (again, for instances hitting the time limit, this is just the number of nodes explored within the limit). In all the verification times the solution of (12)

	1-IP	1-IP +HG	1-IP +Fix	1-IP +Fix +2Var
Preprocessing time (seconds)	0.0	10.8	7.0	15.2
Verification time (seconds)	211.7	258.9	40.6	25.5
# of solved instances (veri.)	376	370	388	431
# of nodes (veri.)	9228.2	9501.0	78.0	7.3
Optimization time (seconds)	298.2	352.0	62.6	36.7
# of solved instances (opt.)	347	347	359	406
# of nodes (opt.)	16 389.5	16 540.8	241.1	31.4
Optimality gap	22.5 %	24.4 %	19.2 %	4.8 %

Table 6: Other metrics of IP methods based on 1-IP (445 instances)

Norm	Many-IP	1-IP	1-IP +HG	1-IP +Fix	1-IP +Fix +2Var
l_∞	0.018	0.026	0.020	0.025	0.032
l_2	0.029	0.051	0.051	0.247	0.312

Table 7: Maximum ϵ under which $\bar{\mathbf{x}}$ is ϵ -verified for l_∞ norm and l_2 norm

is terminated as soon as the verification question is answered (e.g., if the upper bound becomes non-positive or the lower bound becomes positive). To provide further insight into the quality of these formulations we also attempted to solve (12) without terminating once the verification question is answered. The results of this experiment are reported in the rows ‘Optimization time’, ‘(# of solved instances (opti.)’, ‘(# of nodes (opti.))’, which are defined analogously as the verification results. Row ‘optimality gap’ presents the arithmetic mean of relative optimality gaps obtained in solving (12) without the early termination, where the relative optimality gap is defined as $(\bar{z} - z^*)/\bar{z}_{UB}$, where \bar{z} is the best bound for the optimal objective value of (12) obtained by the method. A shift is set to 1 in every shifted geometric mean. These results confirm the significant time reduction obtained using variable fixing and the proposed two-variable inequalities, and indicate that this reduction is obtained thanks to the dramatic reduction in the number of branch-and-bound nodes required to either answer the verification problem or to solve the optimization problem. The results also reinforce that the single-neuron inequalities used in 1-IP+HG do not yield improvement.

All results so far have been reported for the case in which the maximum perturbation from $\bar{\mathbf{x}}$ is defined using the l_1 norm. To illustrate the versatility of the IP approach, we also conducted analogous experiments using the l_∞ and l_2 norm. The detailed results of these experiments are presented in Appendix A.5, and we present a summary here. Table 7 presents the arithmetic mean of the maximum ϵ over ten feature vectors $\bar{\mathbf{x}}$ for each IP method on Network 5. Once gain, Many-IP achieves the smallest maximum ϵ in both cases. Method 1-IP+HG again yields smaller maximum ϵ than 1-IP in the case of $p = \infty$ and the same in the case of $p = 2$. 1-IP+Fix yields similar maximum ϵ to 1-IP in the case of $p = \infty$ and larger maximum ϵ than 1-IP in the case of $p = 2$. Thus, it seems that generating variable fixings alone does not improve BNN verification by the IP methods in the case of $p = \infty$, but it does in the case of $p = 2$. 1-IP+Fix+2Var yields the largest maximum ϵ in both cases, showing that generating layerwise derived valid inequalities enables verifying the BNN against a higher range of ϵ in the case of $p = \infty$ and $p = 2$.

5 Conclusion

In this paper, we investigate an IP method that exploits the technique for obtaining a linear objective and a technique for generating layerwise derived valid inequalities to solve the BNN verification problem. Our computational study shows that our IP method verifies BNNs against a higher range of input perturbation than existing IP methods. The technique for obtaining a linear objective leads to solving the BNN verification problem faster by solving a single IP problem instead of multiple IP problems for each alternative class. The technique for generating layerwise derived valid inequalities enables more efficient BNN verification by yielding smaller LP gaps at the expense of solving IP subproblems involving a single layer.

One future direction on IP methods for the BNN verification problem is investigating valid inequalities that involve more than two variables, as the inequalities we investigated contain at most two variables. Another future direction related to the BNN verification problem is counterfactual explanations for BNNs. As discussed in Section 1, these counterfactual explanations can be obtained by solving the BNN verification problem for various input perturbations. Thus, a possible direction of research for counterfactual explanations for BNNs is to consider whether information from these related instances can be shared to accelerate the overall process. For example, it may be possible to reduce the number of candidates for layerwise derived valid inequalities to check.

References

- Amir G, Wu H, Barrett C, Katz G (2021) An smt-based approach for verifying binarized neural networks. Groote JF, Larsen KG, eds., *Tools and Algorithms for the Construction and Analysis of Systems*, 203–222 (Springer).
- Balas E (1985) Disjunctive programming and a hierarchy of relaxations for discrete optimization problems. *SIAM Journal on Algebraic Discrete Methods* 6(3):466–486.
- Bernardelli AM, Gualandi S, Lau HC, Milanesi S (2023) The bemi stardust: A structured ensemble of binarized neural networks. Sellmann M, Tierney K, eds., *Learning and Intelligent Optimization*, 443–458 (Springer).
- Contardo C, Fukasawa R, Rousseau LM, Vidal T (2024) Optimal counterfactual explanations for k-nearest neighbors using mathematical optimization and constraint programming. Basu A, Mahjoub AR, Salazar González JJ, eds., *Combinatorial Optimization*, 318–331 (Springer).
- Fischetti M, Jo J (2018) Deep neural networks and mixed integer linear optimization. *Constraints* 23(3):296–309.
- Geiger L, Team P (2020) Larq: An open-source library for training binarized neural networks. *Journal of Open Source Software* 5(45):1746.
- Han S, Gómez A (2021) Single-neuron convexification for binarized neural networks. <https://optimization-online.org/wp-content/uploads/2021/05/8419.pdf>, accessed May 27, 2021.
- Hubara I, Courbariaux M, Soudry D, El-Yaniv R, Bengio Y (2016) Binarized neural networks. Lee D, Sugiyama M, von Luxburg U, Guyon I, Garnett R, eds., *Adv. in Neural Information Processing Systems*, volume 29, 4107–4115 (Curran Associates, Inc.).
- Ivashchenko M, Choi SW, Nguyen LV, Tran HD (2023) Verifying binary neural networks on continuous input space using star reachability. *The IEEE International Conf. on Formal Methods in Software Engineering*, 7–17 (IEEE).
- Jia K, Rinard M (2020) Efficient exact verification of binarized neural networks. Larochelle H, Ranzato M, Hadsell R, Balcan MF, Lin HT, eds., *Adv. in Neural Information Processing Systems*, volume 33, 1782–1795 (Curran Associates, Inc.).
- Khalil EB, Gupta A, Dilkina B (2019) Combinatorial attacks on binarized neural networks. *International Conf. on Learning Representations*, URL <https://openreview.net/forum?id=S11TEh09FQ>.

- Kovácsnai G, Gajdár K, Narodytska N (2021) Portfolio solver for verifying binarized neural networks. *Annales Mathematicae et Informaticae* 53:183–200.
- Kung J, Zhang D, van der Wal G, Chai S, Mukhopadhyay S (2018) Efficient object detection using embedded binarized neural networks. *Journal of Signal Processing Systems* 90(6):877–890.
- Lubczyk D, Neto J (2024) Neuron pairs in binarized neural networks robustness verification via integer linear programming. Basu A, Mahjoub AR, Salazar González JJ, eds., *Combinatorial Optimization*, 305–317 (Springer).
- Ma Y, Xiong H, Hu Z, Ma L (2019) Efficient super resolution using binarized neural network. *The IEEE/CVF Conf. on Computer Vision and Pattern Recognition Workshops*, 694–703 (IEEE).
- Martinez B, Yang J, Bulat A, Tzimiropoulos G (2020) Training binary neural networks with real-to-binary convolutions. *International Conf. on Learning Representations*.
- McDanel B, Teerapittayanon S, Kung HT (2017) Embedded binarized neural networks. Gunningberg P, Voigt T, Mottola L, Lu C, eds., *Proc. of the International Conf. on Embedded Wireless Systems and Networks*, 168–173 (Junction Publishing).
- Mothilal R, Sharma A, Tan C (2020) Explaining machine learning classifiers through diverse counterfactual explanations. Hildebrandt M, Castillo C, Celis E, Ruggieri S, Taylor L, Zanfir-Fortuna G, eds., *Conf. on Fairness, Accountability, and Transparency*, 607–617 (ACM).
- Narodytska N, Kasiviswanathan S, Ryzhyk L, Sagiv M, Walsh T (2018) Verifying properties of binarized deep neural networks. McIlraith S, Weinberger K, eds., *Proc. of the AAAI Conf. on Artificial Intelligence*, volume 32, 6615–6624 (AAAI).
- Shih A, Darwiche A, Choi A (2019) Verifying binarized neural networks by angluin-style learning. Janota M, Lynce I, eds., *Theory and Applications of Satisfiability Testing*, 354–370 (Springer).
- Shridhar K, Jain H, Agarwal A, Kleyko D (2020) End to end binarized neural networks for text classification. Moosavi NS, Fan A, Shwartz V, Glavaš G, Joty S, Wang A, Wolf T, eds., *Proc. of SustainLP: Workshop on Simple and Efficient Natural Language Processing*, 29–34 (Association for Computational Linguistics).
- Tang W, Hua G, Wang L (2017) How to train a compact binary neural network with high accuracy? Singh S, Markovitch S, eds., *Proc. of the AAAI Conf. on Artificial Intelligence*, volume 31, 2625–2631 (AAAI).
- Toro Icarte R, Illanes L, Castro M, Cire A, McIlraith S, Beck C (2019) Training binarized neural networks using mip and cp. Schiex T, de Givry S, eds., *Principles and Practice of Constraint Programming*, 401–417 (Springer).
- Vivier-Ardisson G, Forel A, Parmentier A, Vidal T (2024) Cf-opt: Counterfactual explanations for structured prediction. <https://arxiv.org/pdf/2405.18293>, accessed May 28, 2024.
- Wachter S, Mittelstadt B, Russell C (2018) Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harvard Journal of Law and Technology* 31(2):841–888.
- Zhang Y, Zhao Z, Chen G, Song F, Chen T (2021) Bdd4bnn: A bdd-based quantitative analysis framework for binarized neural networks. Silva A, Leino KRM, eds., *Computer Aided Verification*, 175–200 (Springer).

A Appendix

A.1 Proof of Lemma 1

We prove following Lemma 1 on the formulation for the layer propagation constraint (2b) introduced in Section 2.1.1 with the definition on LB_i^ℓ , UB_i^ℓ , and R_i^ℓ for $\ell \in [L]$ and $i \in N^\ell$.

Lemma 1. *Each $(\mathbf{x}^0, \mathbf{x}^1) \in X^0 \times \{0, 1\}^{n^1}$ satisfies (2b) for $\ell = 1$ if and only if it satisfies the following inequalities:*

$$2 \sum_{j \in N^0} W_{ij}^1 x_j^0 \geq \left(R_i^1 - \text{LB}_i^1 + \frac{1}{q} \right) x_i^1 + \text{LB}_i^1, \quad \forall i \in N^1, \quad (4)$$

$$2 \sum_{j \in N^0} W_{ij}^1 x_j^0 \leq \left(\text{UB}_i^1 - R_i^1 + \frac{1}{q} \right) x_i^1 + \left(R_i^1 - \frac{1}{q} \right), \quad \forall i \in N^1. \quad (5)$$

For $\ell \in \{2, \dots, L\}$, $(\mathbf{x}^{\ell-1}, \mathbf{x}^\ell) \in \{0, 1\}^{n^{\ell-1}} \times \{0, 1\}^{n^\ell}$ satisfies (2b) if and only if it satisfies the following inequalities:

$$2 \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1} \geq (R_i^\ell - \text{LB}_i^\ell + 1) x_i^\ell + \text{LB}_i^\ell, \quad \forall i \in N^\ell, \quad (6)$$

$$2 \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1} \leq (\text{UB}_i^\ell - R_i^\ell + 1) x_i^\ell + (R_i^\ell - 1), \quad \forall i \in N^\ell. \quad (7)$$

. Proof of Lemma 1. The following constraints are equivalent to (2b):

$$\begin{aligned} x_i^\ell = 1 &\Rightarrow a_i^\ell(\mathbf{x}^{\ell-1}) \geq 0 \\ &\Leftrightarrow 2 \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1} \geq \sum_{j \in N^{\ell-1}} W_{ij}^\ell - b_i^\ell, \quad \forall \ell \in [L], i \in N^\ell, \\ x_i^\ell = 0 &\Rightarrow a_i^\ell(\mathbf{x}^{\ell-1}) < 0 \\ &\Leftrightarrow 2 \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1} < \sum_{j \in N^{\ell-1}} W_{ij}^\ell - b_i^\ell, \quad \forall \ell \in [L], i \in N^\ell. \end{aligned}$$

For $i \in N^1$ and $j \in N^0$, $W_{ij}^1 x_j^0$ is a multiple of $\frac{1}{q}$. Also, $R_i^1 + \frac{1}{q}$ is the smallest multiple of $\frac{2}{q}$ not smaller than $\sum_{j \in N^0} W_{ij}^1 - b_i^1$, and $R_i^1 - \frac{1}{q}$ is the largest multiple of $\frac{2}{q}$ smaller than $\sum_{j \in N^0} W_{ij}^1 - b_i^1$. Hence, the above constraints can be written as follows for $\ell = 1$:

$$\begin{aligned} x_i^1 = 1 &\Rightarrow 2 \sum_{j \in N^0} W_{ij}^1 x_j^0 \geq R_i^1 + \frac{1}{q}, \quad \forall i \in N^1, \\ x_i^1 = 0 &\Rightarrow 2 \sum_{j \in N^0} W_{ij}^1 x_j^0 \leq R_i^1 - \frac{1}{q}, \quad \forall i \in N^1. \end{aligned}$$

For $\ell \in \{2, \dots, L\}$, $i \in N^\ell$, and $j \in N^{\ell-1}$, $W_{ij}^\ell x_j^{\ell-1}$ is an integer. Also, $R_i^\ell + 1$ is the smallest even integer not smaller than $\sum_{j \in N^{\ell-1}} W_{ij}^\ell - b_i^\ell$, and $R_i^\ell - 1$ is the largest even integer smaller than $\sum_{j \in N^{\ell-1}} W_{ij}^\ell - b_i^\ell$. Hence, the above constraints can be written as follows for $\ell \in \{2, \dots, L\}$:

$$\begin{aligned} x_i^\ell = 1 &\Rightarrow 2 \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1} \geq R_i^\ell + 1, \quad \forall i \in N^\ell, \\ x_i^\ell = 0 &\Rightarrow 2 \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1} \leq R_i^\ell - 1, \quad \forall i \in N^\ell. \end{aligned}$$

For $\ell \in [L]$, $i \in N^\ell$, and $j \in N^{\ell-1}$, $W_{ij}^\ell \in \{-1, 0, 1\}$ and $x_j^{\ell-1} \in [0, 1]$. Hence, LB_i^ℓ is a lower bound for $2 \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1}$ because it is minus two times the number of -1 in $\{W_{ij}^\ell : j \in N^{\ell-1}\}$. Also, UB_i^ℓ is an upper bound for $2 \sum_{j \in N^{\ell-1}} W_{ij}^\ell x_j^{\ell-1}$ because it is two times the number of 1 in $\{W_{ij}^\ell : j \in N^{\ell-1}\}$.

By exploiting this lower bound and upper bound, it can be concluded that $(\mathbf{x}^0, \mathbf{x}^1) \in X^0 \times \{0, 1\}^{n^1}$ satisfies (2b) if and only if it satisfies (4) and (5), and $(\mathbf{x}^{\ell-1}, \mathbf{x}^\ell) \in \{0, 1\}^{n^{\ell-1}} \times \{0, 1\}^{n^\ell}$ satisfies (2b) if and only if it satisfies (6) and (7) for $\ell \in \{2, \dots, L\}$. \square

A.2 Proof of Lemma 2

. Proof. First, from a feasible solution $(\hat{\mathbf{x}}^0, \dots, \hat{\mathbf{x}}^L, \hat{t})$ to (2) where t is considered as a decision variable, we obtain a feasible solution to (12) with the same objective value. From $(\hat{\mathbf{x}}^0, \dots, \hat{\mathbf{x}}^L, \hat{t})$, $\hat{\mathbf{y}}$ is defined as $q\hat{\mathbf{x}}^0$. Also, \hat{z}_t is defined as $1_{\{\hat{t}\}}(t)$, and \hat{v}_{ti} is defined as $\hat{z}_t \hat{x}_i^L$.

By Lemma 1, $(\hat{\mathbf{x}}^0, \dots, \hat{\mathbf{x}}^L, \hat{\mathbf{y}}, \hat{\mathbf{z}}, \hat{\mathbf{v}})$ satisfies (4)-(5) and (6)-(7) for $\ell \in \{2, \dots, L\}$. By the definition of $\hat{\mathbf{y}}$, $(\hat{\mathbf{x}}^0, \dots, \hat{\mathbf{x}}^L, \hat{\mathbf{y}}, \hat{\mathbf{z}}, \hat{\mathbf{v}})$ satisfies (8)-(9) and (12g). By the definition of $\hat{\mathbf{z}}$ and $\hat{\mathbf{v}}$, $(\hat{\mathbf{x}}^0, \dots, \hat{\mathbf{x}}^L, \hat{\mathbf{y}}, \hat{\mathbf{z}}, \hat{\mathbf{v}})$ satisfies (12b)-(12c) and (12h)-(12i). Also, $(\hat{\mathbf{x}}^0, \dots, \hat{\mathbf{x}}^L, \hat{\mathbf{y}}, \hat{\mathbf{z}}, \hat{\mathbf{v}})$ satisfies (12d) because $\hat{\mathbf{z}}$ and $\hat{\mathbf{x}}^L$ are binary, so $(\hat{\mathbf{x}}^0, \dots, \hat{\mathbf{x}}^L, \hat{\mathbf{y}}, \hat{\mathbf{z}}, \hat{\mathbf{v}})$ is a feasible solution to (12).

The definitions of $\hat{\mathbf{z}}$ and $\hat{\mathbf{v}}$ implies $\hat{z}_t = 0$ for $t \neq \hat{t}$, $\hat{z}_{\hat{t}} = 1$, and $\hat{v}_{ti} = \hat{z}_t \hat{x}_i^L$. With this implication, the objective value of $(\hat{\mathbf{x}}^0, \dots, \hat{\mathbf{x}}^L, \hat{t})$ in (2) is same as the objective value of $(\hat{\mathbf{x}}^0, \dots, \hat{\mathbf{x}}^L, \hat{\mathbf{y}}, \hat{\mathbf{z}}, \hat{\mathbf{v}})$ in (12) by following steps:

$$\begin{aligned}
a_{\hat{t}}^{L+1}(\hat{\mathbf{x}}^L) - a_{\bar{t}}^{L+1}(\hat{\mathbf{x}}^L) &= \sum_{t \in N^{L+1} \setminus \{\hat{t}\}} \hat{z}_t (a_t^{L+1}(\hat{\mathbf{x}}^L) - a_{\bar{t}}^{L+1}(\hat{\mathbf{x}}^L)) \\
&= \sum_{t \in N^{L+1} \setminus \{\hat{t}\}} \hat{z}_t \left(\sum_{i \in N^L} (W_{ti}^L - W_{\bar{t}i}^L) (2\hat{x}_i^L - 1) + (b_t^L - b_{\bar{t}}^L) \right) \\
&= \sum_{t \in N^{L+1} \setminus \{\hat{t}\}} \sum_{i \in N^L} 2(W_{ti}^L - W_{\bar{t}i}^L) \hat{z}_t \hat{x}_i^L \\
&\quad + \sum_{t \in N^{L+1} \setminus \{\hat{t}\}} \left(- \sum_{i \in N^L} (W_{ti}^L - W_{\bar{t}i}^L) + (b_t^L - b_{\bar{t}}^L) \right) \hat{z}_t \\
&= \sum_{t \in N^{L+1} \setminus \{\hat{t}\}} \sum_{i \in N^L} 2(W_{ti}^L - W_{\bar{t}i}^L) \hat{v}_{ti} \\
&\quad + \sum_{t \in N^{L+1} \setminus \{\hat{t}\}} \left(- \sum_{i \in N^L} (W_{ti}^L - W_{\bar{t}i}^L) + (b_t^L - b_{\bar{t}}^L) \right) \hat{z}_t.
\end{aligned} \tag{22}$$

Next, from a feasible solution $(\hat{\mathbf{x}}^0, \dots, \hat{\mathbf{x}}^L, \hat{\mathbf{y}}, \hat{\mathbf{z}}, \hat{\mathbf{v}})$ to (12), we obtain a feasible solution to (2) with the same objective value where t is considered as a decision variable. From $(\hat{\mathbf{x}}^0, \dots, \hat{\mathbf{x}}^L, \hat{\mathbf{y}}, \hat{\mathbf{z}}, \hat{\mathbf{v}})$, \hat{t} is defined as t satisfying $\hat{z}_t = 1$, whose uniqueness is guaranteed by (12b) and (12h).

By (4)-(7) and Lemma 1, $(\hat{\mathbf{x}}^0, \dots, \hat{\mathbf{x}}^L, \hat{t})$ satisfies the layer propagation constraints (2b). By (8)-(10), (12e), and (12g), $(\hat{\mathbf{x}}^0, \dots, \hat{\mathbf{x}}^L, \hat{t})$ satisfies the input perturbation constraint (2c). Hence, $(\hat{\mathbf{x}}^0, \dots, \hat{\mathbf{x}}^L, \hat{t})$ is a feasible solution to (2).

By (22), the objective value of $(\hat{\mathbf{x}}^0, \dots, \hat{\mathbf{x}}^L, \hat{\mathbf{y}}, \hat{\mathbf{z}}, \hat{\mathbf{v}})$ in (12) is same as the objective value of $(\hat{\mathbf{x}}^0, \dots, \hat{\mathbf{x}}^L, \hat{t})$ in (2). The first equality holds because $\hat{z}_t = 0$ for $t \neq \hat{t}$, and $\hat{z}_{\hat{t}} = 1$. The last equality holds because $\hat{v}_{ti} = \hat{z}_t \hat{x}_i^L$. For $t \neq \hat{t}$, $\hat{v}_{ti} \geq 0$ by (12i), and $\hat{v}_{ti} \leq \hat{z}_t = 0$ by (12d), so $\hat{v}_{ti} = 0 = \hat{z}_t \hat{x}_i^L$. Also, $\hat{v}_{\hat{t}i} = \hat{z}_{\hat{t}} \hat{x}_i^L$ by following steps:

$$\begin{aligned}
\hat{v}_{\hat{t}i} &= \sum_{t \in N^{L+1} \setminus \{\hat{t}\}} \hat{v}_{ti} && \text{(For all } t \text{ other than } \hat{t}, \hat{v}_{ti} = 0) \\
&= \hat{x}_i^L && \text{(By (12c))} \\
&= \hat{z}_{\hat{t}} \hat{x}_i^L.
\end{aligned}$$

For these reasons, (12) is equivalent to (2). \square

A.3 Proof of Convex Hull Characterization for a Single Neuron

We prove Theorem 3 on the characterization for the convex hull (13) for a single neuron introduced in Section 2.3, which is an extension of the main result in Han and Gómez (2021). To simplify the

statement of this theorem, $\ell \in \{2, \dots, L\}$ and $i \in N^\ell$ are fixed, and the following notations are defined for $j \in N^{\ell-1}$ with LB_i^ℓ , UB_i^ℓ , and R_i^ℓ defined in Section 2.1.1:

$$\begin{aligned} \mathbf{x} &:= \mathbf{x}^{\ell-1}, \\ u &:= x_i^\ell, \\ w_j &:= W_{ij}^\ell, \\ \text{LB} &:= \text{LB}_i^\ell = \sum_{j \in N^{\ell-1}} (W_{ij}^\ell - |W_{ij}^\ell|), \\ \text{UB} &:= \text{UB}_i^\ell = \sum_{j \in N^{\ell-1}} (W_{ij}^\ell + |W_{ij}^\ell|), \\ R &:= R_i^\ell = 2 \left\lceil \frac{\sum_{j \in N^{\ell-1}} W_{ij}^\ell - b_i^\ell}{2} \right\rceil - 1. \end{aligned}$$

Then, this theorem can be restated as follows:

Theorem 3 (Restated) *The set of $(\mathbf{x}, u) \in [0, 1]^{n^{\ell-1}} \times [0, 1]$ satisfying the following inequalities for all $J \subset \{j \in N^{\ell-1} : w_j \neq 0\}$ is (13):*

$$\sum_{j \in J} (w_j(2x_j - 1) - (2u - 1)) \geq (R - \text{UB} + 1)u, \quad (23a)$$

$$\sum_{j \in J} (w_j(2x_j - 1) - (2u - 1)) \leq (R - \text{LB} - 1)(-u + 1). \quad (23b)$$

. Proof. Consider the following sets:

$$\begin{aligned} X^+ &= \{(\mathbf{x}, 1) : \mathbf{x} \in \{0, 1\}^{n^{\ell-1}}, a_i^\ell(\mathbf{x}) \geq 0\}, \\ X^- &= \{(\mathbf{x}, 0) : \mathbf{x} \in \{0, 1\}^{n^{\ell-1}}, a_i^\ell(\mathbf{x}) < 0\}. \end{aligned}$$

By the proof of Lemma 1, X^+ and X^- can be written as follows:

$$\begin{aligned} X^+ &= \left\{ (\mathbf{x}, 1) : \mathbf{x} \in \{0, 1\}^{n^{\ell-1}}, \sum_{\substack{j \in N^{\ell-1} \\ w_j \neq 0}} w_j x_j \geq \frac{R+1}{2} \right\}, \\ X^- &= \left\{ (\mathbf{x}, 0) : \mathbf{x} \in \{0, 1\}^{n^{\ell-1}}, \sum_{\substack{j \in N^{\ell-1} \\ w_j \neq 0}} w_j x_j \leq \frac{R-1}{2} \right\}. \end{aligned}$$

By exploiting total unimodularity of the constraints defining X^+ and X^- arising from $w_j \in \{-1, 0, 1\}$ and $\frac{R+1}{2} \in \mathbb{Z}$, $\text{conv}(X^+)$ and $\text{conv}(X^-)$ can be obtained as follows:

$$\begin{aligned} \text{conv}(X^+) &= \left\{ (\mathbf{x}, 1) : \mathbf{x} \in [0, 1]^{n^{\ell-1}}, \sum_{\substack{j \in N^{\ell-1} \\ w_j \neq 0}} w_j x_j \geq \frac{R+1}{2} \right\}, \\ \text{conv}(X^-) &= \left\{ (\mathbf{x}, 0) : \mathbf{x} \in [0, 1]^{n^{\ell-1}}, \sum_{\substack{j \in N^{\ell-1} \\ w_j \neq 0}} w_j x_j \leq \frac{R-1}{2} \right\}. \end{aligned}$$

These convex hulls can be used to describe (13) as follows where the last statement holds because of compactness of $\text{conv}(X^+)$ and $\text{conv}(X^-)$ arising from finiteness of X^+ and X^- :

$$\begin{aligned}\text{conv}(X^+ \cup X^-) &= \text{conv}(\text{conv}(X^+) \cup \text{conv}(X^-)) \\ &= \text{clconv}(\text{conv}(X^+) \cup \text{conv}(X^-)).\end{aligned}$$

By Balas (1985)'s work on disjunctive programming, $(\mathbf{x}, u) \in \mathbb{R}^{n^{\ell-1}} \times \mathbb{R}$ is in (13) if and only if there exist $(\mathbf{x}^+, u^+), (\mathbf{x}^-, u^-) \in \mathbb{R}^{n^{\ell-1}} \times \mathbb{R}$ and $\lambda \in [0, 1]$ satisfying the following inequalities:

$$\begin{aligned}\mathbf{x} &= \mathbf{x}^+ + \mathbf{x}^-, \\ u &= u^+ + u^-, \\ 0 &\leq \mathbf{x}^+ \leq \lambda \cdot \mathbf{1}, \\ u^+ &= \lambda, \\ 2 \sum_{\substack{j \in N^{\ell-1} \\ w_j \neq 0}} w_j x_j^+ &\geq (R+1)\lambda, \\ 0 &\leq \mathbf{x}^- \leq (1-\lambda) \cdot \mathbf{1}, \\ u^- &= 0, \\ 2 \sum_{\substack{j \in N^{\ell-1} \\ w_j \neq 0}} w_j x_j^- &\leq (R-1)(1-\lambda).\end{aligned}$$

Existence of $(\mathbf{x}^+, u^+), (\mathbf{x}^-, u^-) \in \mathbb{R}^{n^{\ell-1}} \times \mathbb{R}$ and $\lambda \in [0, 1]$ satisfying the above inequalities is identical to existence of $\mathbf{x}^+ \in \mathbb{R}^{n^{\ell-1}}$ satisfying the following inequalities, which are provided by substituting u^- with 0, u^+ and λ with u , and \mathbf{x}^- with $\mathbf{x} - \mathbf{x}^+$:

$$-\frac{u}{2} \cdot \mathbf{1} \leq \mathbf{x}^+ - \frac{u}{2} \cdot \mathbf{1} \leq \frac{u}{2} \cdot \mathbf{1}, \quad (24)$$

$$-\left(\frac{1-u}{2}\right) \cdot \mathbf{1} \leq \mathbf{x} - \mathbf{x}^+ - \left(\frac{1-u}{2}\right) \cdot \mathbf{1} \leq \left(\frac{1-u}{2}\right) \cdot \mathbf{1}, \quad (25)$$

$$2 \sum_{\substack{j \in N^{\ell-1} \\ w_j \neq 0}} w_j x_j^+ \geq (R+1)u, \quad (26)$$

$$2 \sum_{\substack{j \in N^{\ell-1} \\ w_j \neq 0}} w_j x_j^+ \geq 2 \sum_{\substack{j \in N^{\ell-1} \\ w_j \neq 0}} w_j x_j - (R-1)(1-u). \quad (27)$$

For j satisfying $w_j \neq 0$, (24)-(25) are equivalent to the following inequalities because $w_j \in \{-1, 1\}$:

$$\begin{aligned}-\frac{u}{2} \cdot \mathbf{1} &\leq w_j \left(\mathbf{x}^+ - \frac{u}{2} \right) \cdot \mathbf{1} \leq \frac{u}{2} \cdot \mathbf{1}, \\ -\frac{(1-u)}{2} \cdot \mathbf{1} &\leq w_j \left(\mathbf{x} - \mathbf{x}^+ - \left(\frac{1-u}{2} \right) \right) \cdot \mathbf{1} \leq \frac{(1-u)}{2} \cdot \mathbf{1}.\end{aligned}$$

By replacing $w_j x_j^+$ with its maximum attained from the above inequalities in (26)-(27) and employing Fourier-Motzkin Elimination to (24)-(25) to remove $\mathbf{x}^+ - \frac{u}{2}$, existence of $\mathbf{x}^+ \in \mathbb{R}^{n^{\ell-1}}$ satisfying

(24)-(27) is equivalent to whether the following inequalities are satisfied:

$$\begin{aligned} \max\left(-\frac{u}{2}, x_j + \frac{u}{2} - 1\right) &\leq \min\left(\frac{u}{2}, x_j - \frac{u}{2}\right), \quad \forall j \in N^{\ell-1}, \\ \sum_{\substack{j \in N^{\ell-1} \\ w_j \neq 0}} \min((w_j + 1)u, 2w_j x_j + (-w_j + 1)(1 - u)) &\geq (R + 1)u, \\ \sum_{\substack{j \in N^{\ell-1} \\ w_j \neq 0}} \min((w_j + 1)u, 2w_j x_j + (-w_j + 1)(1 - u)) &\geq 2 \sum_{\substack{j \in N^{\ell-1} \\ w_j \neq 0}} w_j x_j - (R - 1)(1 - u). \end{aligned}$$

These inequalities are satisfied if and only if the following inequalities are satisfied because $|w_j| = 1$ if $w_j \neq 0$, $\sum_{\substack{j \in N^{\ell-1} \\ w_j \neq 0}} (-w_j + 1) = \sum_{j \in N^{\ell-1}} (-w_j + |w_j|) = -\text{LB}$, and $\sum_{\substack{j \in N^{\ell-1} \\ w_j \neq 0}} (w_j + 1) = \sum_{j \in N^{\ell-1}} (w_j + |w_j|) = \text{UB}$:

$$\begin{aligned} 0 &\leq x_j \leq 1, \quad \forall j \in N^{\ell-1}, \\ 0 &\leq u \leq 1, \\ \sum_{\substack{j \in N^{\ell-1} \\ w_j \neq 0}} \min(w_j(2x_j - 1), 2u - 1) &\geq (R - \text{LB} + 1)u - \sum_{j \in N^{\ell-1}} |w_j|, \\ \sum_{\substack{j \in N^{\ell-1} \\ w_j \neq 0}} \max(w_j(2x_j - 1), 2u - 1) &\leq (\text{UB} - R + 1)u + R - \sum_{j \in N^{\ell-1}} w_j - 1. \end{aligned}$$

If $w_j \neq 0$, $|w_j| = 1$, so $|\{j \in N^{\ell-1} : w_j \neq 0\}| = \sum_{j \in N^{\ell-1}} |w_j|$. With this equality, by replacing min and max in the last two inequalities with linear expressions involving $J \subset \{j \in N^{\ell-1} : w_j \neq 0\}$, the set of (\mathbf{x}, u) satisfying the last two inequalities can be written as the set of (\mathbf{x}, u) satisfying the following linear inequalities for all $J \subset \{j \in N^{\ell-1} : w_j \neq 0\}$:

$$\begin{aligned} \sum_{j \in J} (w_j(2x_j - 1) - (2u - 1)) &\geq (R - \text{LB} + 1)u - \sum_{j \in N^{\ell-1}} |w_j| - \left(\sum_{j \in N^{\ell-1}} |w_j|\right)(2u - 1), \\ \sum_{j \in J} (w_j(2x_j - 1) - (2u - 1)) &\leq (\text{UB} - R + 1)u + R - \sum_{j \in N^{\ell-1}} w_j - 1 - \left(\sum_{j \in N^{\ell-1}} |w_j|\right)(2u - 1). \end{aligned}$$

These inequalities are equivalent to (23a) and (23b) because $\sum_{j \in N^{\ell-1}} |w_j| = \frac{\text{UB} - \text{LB}}{2}$ and $\sum_{j \in N^{\ell-1}} (w_j - |w_j|) = \text{LB}$, so (13) is the set of $(\mathbf{x}, u) \in [0, 1]^{N^{\ell-1}} \times [0, 1]$ satisfying (23) for all $J \subset \{j \in N^{\ell-1} : w_j \neq 0\}$. \square

A.4 Implementation Details for Infinity-norm and 2-norm

We explain implementation details for the case that the maximum perturbation from $\bar{\mathbf{x}}$ is defined using an ℓ_∞ norm or ℓ_2 norm. These details are about how to solve the IP subproblem (17) without an IP solver in the case of $\ell = 1$ and how to implement the input perturbation constraint (10).

We first consider solving (17) for the case of layer $\ell = 1$ and $p = \infty$. In this case, the optimal solution is obtained by setting x_j^0 to $\min(\bar{x}_j + \epsilon, 1)$ for j satisfying $c_i W_{ij}^1 = 1$, setting x_j^0 to $\max(\bar{x}_j - \epsilon, 0)$ for j satisfying $c_i W_{ij}^1 = -1$, and setting x_j^0 to \bar{x}_j for j satisfying $c_i W_{ij}^1 = 0$. All constraints defining X^0 are of the form $|x_j^0 - \bar{x}_j| \leq \epsilon$, so this solution is feasible, and it immediate that no other solution can yield a better objective value.

We next consider solving (17) for the case of layer $\ell = 1$ and $p = 2$. First, observe that given a feasible solution $\hat{\mathbf{x}}^0 \in \frac{1}{q}\mathbb{Z}_+^{n^0} \cap [0, 1]^{n^0}$ to (17), if for some $j \in N^0$ it holds that $c_i W_{ij}^1 (\hat{x}_j^0 - \bar{x}_j) \leq 0$, then another feasible solution with the same or better objective value can be obtained by replacing \hat{x}_j^0 with \bar{x}_j . Hence, we can restrict our search for an optimal solution of (17) to solutions $\hat{\mathbf{x}}^0$ that satisfy:

$$\begin{aligned} \hat{x}_j^0 &\geq \bar{x}_j, & \text{for } j \text{ such that } c_i W_{ij}^1 &= 1, \\ \hat{x}_j^0 &\leq \bar{x}_j, & \text{for } j \text{ such that } c_i W_{ij}^1 &= -1, \\ \hat{x}_j^0 &= \bar{x}_j, & \text{for } j \text{ such that } W_{ij}^1 &= 0. \end{aligned}$$

Let $N_0^+ = \{j \in N_0 : W_{ij} \neq 0\}$. Thus, we can restrict our attention to solutions defined by a vector $z \in \{0, 1, \dots, q\}^{N_0^+}$ as follows:

$$\hat{x}_j^0(z) = \begin{cases} \min(\bar{x}_j + \frac{z_j}{q}, 1) & \text{if } c_i W_{ij}^1 = 1, \\ \max(\bar{x}_j - \frac{z_j}{q}, 0) & \text{if } c_i W_{ij}^1 = -1, \\ \bar{x}_j & \text{if } W_{ij}^1 = 0. \end{cases}$$

Observe that the objective in (17) and the expression $\|\hat{\mathbf{x}}(z) - \bar{\mathbf{x}}\|_2$ are monotone increasing in z , and hence an optimal solution will be such that increasing z_j for any $j \in N_0^+$ would be infeasible. Next, for $z \in \{0, 1, \dots, q\}^{N_0^+}$, define the set

$$N_0^S(z) = \left\{ j \in N_0^+ : (c_i W_{ij}^1 = 1 \text{ and } \hat{x}_j^0(z) \leq 1 - 1/q) \text{ or } (c_i W_{ij}^1 = -1 \text{ and } \hat{x}_j^0(z) \geq 1/q) \right\}.$$

If there exists $z \in \{0, 1, \dots, q\}^{N_0^+}$ in which $\hat{\mathbf{x}}(z)$ is feasible to (17) and $N_0^S(z) = \emptyset$, then $\hat{\mathbf{x}}(z)$ is optimal to (17) since this solution obtains the best possible objective. Otherwise, we claim that there exists an optimal solution $\hat{\mathbf{x}}(z)$ that satisfies:

$$\max\{z_j : j \in N_0^S(z)\} - \min\{z_j : j \in N_0^S(z)\} \leq 1.$$

Consider a solution that violates this condition. Another feasible solution with the same objective value can be obtained by decreasing z_{j_1} by one and increasing z_{j_2} by one, where $j_1 \in \arg \max\{z_j : j \in N_0^S(z)\}$ and $j_2 \in \arg \min\{z_j : j \in N_0^S(z)\}$. Repeating this process will eventually yield a vector z which satisfies this condition since there only finitely many elements achieving the max and min in the condition, and an element will necessarily be removed from one or the other as long as the max is at least two larger than the min.

These arguments imply that an optimal solution to (17) can be obtained by first finding the maximum m such that the solution $\hat{\mathbf{x}}(z)$ is satisfies $\|\hat{\mathbf{x}}(z) - \bar{\mathbf{x}}\|_2 \leq \epsilon$, where $z_j = m$ for $j \in N_0^+$. Having found this solution (e.g., by binary search), one would then iteratively select some $j \in N_0^S(z)$ and increase z_j , and repeat (without selecting any j more than once) until no more can be increased without violating $\|\hat{\mathbf{x}}(z) - \bar{\mathbf{x}}\|_2 \leq \epsilon$.

Finally, we describe how the constraint (10) is formulated when using a MIP solver to solve the verification problem in the case of $p = 2$. We define a decision variable u_j to represent $|x_j^0 - \bar{x}_j|$ for each $j \in N^0$. The following inequalities are added as constraints because (10) is satisfied if and only if there exists $\mathbf{u} \in \mathbb{R}_+^{n^0}$ satisfying the following inequalities:

$$\begin{aligned} u_j &\geq x_j^0 - \bar{x}_j, & \forall j \in N^0, \\ u_j &\geq -x_j^0 + \bar{x}_j, & \forall j \in N^0, \end{aligned}$$

Norm	# of instances	Many-IP	1-IP
ℓ_∞	47	143.1(16)	78.1(6)
ℓ_2	17	502.9(12)	311.6(2)

Table 8: Verification time (seconds) of Many-IP and 1-IP for ℓ_∞ norm and ℓ_2 norm

	1-IP	1-IP +HG	1-IP +Fix	1-IP +Fix +2Var
Preprocessing time (seconds)	0.0	10.6	7.5	27.4
Verification time (seconds)	98.6	147.2	100.6	44.9
# of solved instances (veri.)	38	35	37	44
# of nodes (veri.)	1297.6	1283.6	1265.3	17.8
Optimization time (seconds)	115.5	178.4	117.5	57.4
# of solved instances (opt.)	37	33	37	42
# of nodes (opt.)	2182.8	2587.9	2133.7	86.4
LP gap	153.4%	156.3%	103.4%	35.2%
Optimality gap	31.8%	46.8%	33.6%	11.0%

Table 9: Metrics of IP methods based on 1-IP for ℓ_∞ norm (49 instances)

$$\sum_{j \in N^0} u_j^2 \leq \epsilon^2.$$

We remark that we found through preliminary empirical study that this formulation technique was computationally superior to the more direct approach of simply formulating the constraint as follows:

$$\sum_{j \in N_0} (x_j^0 - \bar{x}_j)^2 \leq \epsilon^2.$$

A.5 Detailed Computational Results for Infinity-norm and 2-norm

We present more detailed computational results for the case that the maximum perturbation from $\bar{\mathbf{x}}$ is defined using an ℓ_∞ norm or ℓ_2 norm. Test instances for these cases are formed with Network 5 in Table 1, $\bar{\mathbf{x}}$, and ϵ that are chosen as explained in Section 4.1.

We first investigate the time required to use the IP problem (12) (resp. (11) for all t) to solve the the verification problem. The shifted geometric mean (with a shift of 1) of times over all instances tested by both IP method Many-IP and 1-IP is computed for each method and displayed in Table 8 for the case of $p = \infty$ and $p = 2$, along with the number of test instances. For Many-IP, the number in parenthesis is the number of test instances where the time limit is hit in solving (11) for at least one t . For 1-IP, the number in parenthesis is the number of test instances where the time limit is hit in solving (12). We find that 1-IP’s verification times are smaller than Many-IP’s, indicating that our technique for obtaining a linear objective also leads to faster BNN verification in the case of $p = \infty$ and $p = 2$.

Table 9 presents results analogous to those of Table 6 in the main sections, but on instances with $p = \infty$. We find that in this case, neither 1-IP+HG nor 1-IP+Fix yield improvement compared to 1-IP, so generating (14) and variable fixings do not improve BNN verification. Generating variable fixings decreases LP gaps, but this decrease does not lead to more effective BNN verification in

	1-IP	1-IP +HG	1-IP +Fix	1-IP +Fix +2Var
Preprocessing time (seconds)	0.0	35.1	6.7	9.2
Verification time (seconds)	1215.6	1305.9	9.3	10.3
# of solved instances (veri.)	12	12	22	22
# of nodes (veri.)	9844.0	9811.1	0.8	0.7
Optimization time (seconds)	1549.9	1632.5	12.2	12.4
# of solved instances (opt.)	11	11	21	22
# of nodes (opt.)	16 647.4	16 607.7	7.1	5.0
LP gap	165.1 %	165.1 %	12.5 %	2.9 %
Optimality gap	46.5 %	46.6 %	1.8 %	0.0 %

Table 10: Metrics of IP methods based on 1-IP for ℓ_2 norm (22 instances)

	1-IP +Fix	1-IP +Fix +2Var
Preprocessing time (seconds)	7.0	24.0
Verification time (seconds)	66.3	41.5
# of solved instances (veri.)	35	42
# of nodes (veri.)	33.6	4.8
Optimization time (seconds)	82.3	82.9
# of solved instances (opt.)	32	34
# of nodes (opt.)	90.7	75.7
LP gap	51.8 %	24.0 %
Optimality gap	29.5 %	16.9 %

Table 11: Metrics of 1-IP+Fix and 1-IP+Fix+2Var for ℓ_2 norm (49 instances)

the case of $p = \infty$. On the other hand, 1-IP+Fix+2Var yields improvement compared to the other IP methods based on 1-IP. It implies that our technique for generating layerwise derived valid inequalities, mainly two-variable inequalities, also leads to more efficient BNN verification in the case of $p = \infty$.

In the case of $p = 2$, the same metrics are computed over all 22 instances tested by all IP methods based on 1-IP to compare these IP methods. Table 10 shows these metrics. 1-IP+HG does not result in improvement compared to 1-IP, which means that generating (14) also does not improve BNN verification in the case of $p = 2$. 1-IP+Fix and 1-IP+Fix+2Var yield similar improvements compared to 1-IP, which implies that generating variable fixings improves BNN verification in the case of $p = 2$, but on these instances there does not appear to be further improvement from two-variable inequalities.

To better compare 1-IP+Fix and 1-IP+Fix+2Var, we report in Table 11 the results comparing these instances on all 49 instances solved by these two methods (many of these were excluded in Table 10 since they were not solved by 1-IP or 1-IP+HG). From this table we observe that spending more time in generating more layerwise derived valid inequalities, mainly two-variable inequalities, results in reducing LP gaps and solving the verification problem more quickly on these instances.

Network	L	n^1	n^2
1	1	32	*
2	1	64	*
3	1	128	*
4	2	32	32
5	2	64	64
6	2	128	128

Table 12: Networks in additional computational study

A.6 Computational Study on Convex Hull Characterization for a Single Neuron

We pursue an additional computational study to validate our implementation for IP methods exploiting Theorem 3 on convex hull characterization for a single neuron. As we see in in Table 5, the IP method 1-IP+HG employing the valid inequalities presented Theorem 3 does not yield significant decreases in LP gaps compared to the IP method 1-IP. However, using these valid inequalities resulted in smaller LP gaps in Han and Gómez (2021). One possible reason for this discrepancy is that (14) cannot be used for $\ell = 1$ in our computational study because $q = 255$, but these valid inequalities can be used in the computational study of Han and Gómez (2021) because $q = 1$. To examine whether this guess is correct, an additional computational study is conducted on test instances based on ones in Han and Gómez (2021) to assess the impact of exploiting Theorem 3. In this computational study, the IP method Many-IP and Many-IP+HG, which solves the IP problem (11) by employing a constraint generation approach with (14) as 1-IP+HG for each t , are compared because t is fixed in Han and Gómez (2021).

Each test instance consists of a BNN, $\bar{\mathbf{x}}$, and ϵ . Six BNNs pretrained from Han and Gómez (2021) are used in this computational study. These networks are trained on the MNIST training dataset by using Hubara et al. (2016)’s method, which implies $n^0 = 784$ and $n^{L+1} = 10$. The numbers of hidden layers and the number of neurons in hidden layers of each network are reported in Table 12. Feature vectors in the MNIST dataset are scaled to binary vectors by converting coordinates smaller than 128 to 0 and the other coordinates to 1, which indicates $q = 1$. Ten feature vectors are selected for $\bar{\mathbf{x}}$. For each digit from 0 to 9, one feature vector in the MNIST test dataset whose \bar{t} is this digit is randomly chosen. Positive integers from 1 to 5 are used as ϵ .

For each network and ϵ , the arithmetic mean of optimal objective values of the LP relaxation problem of (11) over all $\bar{\mathbf{x}}$ and t (LP value) and the shifted geometric mean of times to solve the verification problem for all t with a shift of 1 over all $\bar{\mathbf{x}}$ (verification time) are computed for Many-IP and Many-IP+HG to compare these IP methods. Table 13 reports LP values and verification times. The number in parenthesis for verification time is the number of $\bar{\mathbf{x}}$ where the time limit is hit in solving (11) for at least one t for each network and ϵ . Many-IP+HG yields smaller LP values than Many-IP as in Han and Gómez (2021), so our additional computational study validates our implementation for IP methods employing the valid inequalities in Theorem 3. In particular, the gap in LP values between Many-IP and Many-IP+HG becomes smaller for larger L and ϵ as in Han and Gómez (2021). However, Many-IP+HG results in larger verification times than Many-IP, which implies that even though using the inequalities in Theorem 3 improves LP relaxation values, it does not reduce BNN verification time.

Network	ϵ	LP value		Verification time (seconds)	
		Many-IP	Many-IP+HG	Many-IP	Many-IP+HG
1	1.0	24.5	6.2	1.2	1.7
	2.0	24.6	6.3	1.3	5.0
	3.0	24.7	7.9	1.4	5.8
	4.0	24.8	7.7	1.4	6.7
	5.0	24.8	7.9	1.5	8.6
2	1.0	50.2	12.9	2.3	3.8
	2.0	50.4	16.9	4.2	9.9
	3.0	50.5	16.4	4.3	13.4
	4.0	50.7	18.5	4.3	16.9
	5.0	50.8	22.9	4.1	17.2
3	1.0	80.7	15.9	4.5	8.6
	2.0	80.9	19.1	9.0	27.5
	3.0	81.2	20.3	9.2	34.1
	4.0	81.4	29.4	9.2	36.6
	5.0	81.6	31.8	9.3	49.3
4	1.0	22.1	4.5	1.2	5.0
	2.0	22.1	11.0	1.4	21.1
	3.0	22.1	14.4	1.5	20.2
	4.0	22.1	16.6	1.4	22.6
	5.0	22.1	18.1	1.5	25.6
5	1.0	29.9	12.7	2.5	8.6
	2.0	29.9	19.9	6.6	53.2
	3.0	29.9	23.6	6.9	47.5
	4.0	29.9	25.7	7.2	68.4
	5.0	29.9	27.0	7.5	84.8
6	1.0	45.0	26.4	5.3	16.3
	2.0	45.0	36.2	28.7	1347.6(3)
	3.0	45.0	40.0	37.1	1312.6(3)
	4.0	45.0	42.2	50.4	1287.8(5)
	5.0	45.0	43.3	64.9	1175.9(4)

Table 13: Metrics of Many-IP and Many-IP+HG