7. Appendix

7.1. Proof of Theorem 1

In this subsection, we prove the soundness of our linear bounds in Theorem 1.

Proof. Define $\boldsymbol{m} = \frac{\boldsymbol{u}+\boldsymbol{l}}{2} = (m_1, \cdots, m_n) \in \mathbb{R}^n$.

Upper linear bound:

Case 1: When $l_i = l_{max} \wedge l_{max} \geq u_j$, we have$f(x_1, \cdots, x_n) = x_i$ and $u(x_1, \cdots, x_n) = x_i - l_i + l_i = x_i$. Then, we have$u(\boldsymbol{x}) \geq f(\boldsymbol{x})$, that is, the upper linear bound of case 1 is sound.

Case 2: When $l_i = l_{max}, u_j > l_i \geq u_k$, we have $f(x_1, \cdots, x_n) = max(x_i, x_j)$ and $u(x_1, \cdots, x_n) = x_i - l_i + \frac{u_j - l_i}{u_j - l_j}(x_j - l_j) + l_i$.

If $f(x_1, \cdots, x_n) = x_i$,

$$\begin{aligned} u(x_1, \cdots, x_n) - x_i =& x_i - l_i + \frac{u_j - l_i}{u_j - l_j}(x_j - l_j) + l_i - x_i \\ =& \frac{u_j - l_i}{u_j - l_j}(x_j - l_i) \\ \geq& \; 0 \end{aligned}$$

If $f(x_1, \cdots, x_n) = x_j$,

$$\begin{aligned} u(x_1, \cdots, x_n) - x_j =& x_i - l_i + \frac{u_j - l_i}{u_j - l_j}(x_j - l_j) + l_i - x_j \\ =& (x_i - l_i) + \frac{u_j - l_i}{u_j - l_j}(x_j - l_j) + l_j - x_j - l_j + l_i \\ =& (x_i - l_i) + \frac{l_j - l_i}{u_j - l_j}(x_j - l_j) - l_j + l_i \\ =& (x_i - l_i) + \frac{u_j - x_j}{u_j - l_j}(l_i - l_j) \\ \geq& 0 \end{aligned}$$

Then, we have $u(\boldsymbol{x}) \geq f(\boldsymbol{x})$, that is, the upper linear bound of case 2 is sound.

Case 3: When $l_i = l_{max} \wedge l_i \neq l_{max} \wedge u_j \geq l_j \geq u_k$, we have $f(x_1, \cdots, x_n) = max(x_i, x_j)$ and $u(x_1, \cdots, x_n) = \frac{u_i - l_j}{u_i - l_i} x_i + x_j + l_j$.

If $f(x_1, \cdots, x_n) = x_i$,

$$\begin{aligned} u(x_1, \cdots, x_n) - x_i =& \frac{u_i - l_j}{u_i - l_i}(x_i - l_i) + (x_j - l_j) + l_j - l_i + l_i - x_i \\ =& (x_i - l_i)(\frac{u_i - l_j}{u_i - l_i} - 1) + (x_j - l_j) + l_j - l_i \\ =& (x_i - l_i)\frac{l_i - l_j}{u_i - l_i} + (x_j - l_j) + l_j - l_i \\ =& (l_j - l_i)(1 - \frac{x_i - l_i}{u_i - l_i}) + (x_j - l_j) \\ =& (l_j - l_i)\frac{u_i - x_i}{u_i - l_i} + (x_j - l_j) \\ \geq& 0 \end{aligned}$$

If $f(x_1, \cdots, x_n) = x_j$,

$$\begin{aligned} u(x_1, \cdots, x_n) - x_j =& \frac{u_i - l_j}{u_i - l_i}(x_i - l_i) + (x_j - l_j) + l_j - x_j \\ =& \frac{u_i - l_j}{u_i - l_i}(x_i - l_i) \\ \geq& 0 \end{aligned}$$

Then, we have $u(\boldsymbol{x}) \geq f(\boldsymbol{x})$, that is, the upper linear bound of case 3 is sound.
Case 4: First, we prove that if $\boldsymbol{u}$ and $\boldsymbol{l}$ do not satisfy case 1,2, and 3, then $u_k > max\{l_i, l_j\}$.
We prove this by contradiction.
We assume that $u_k \leq max\{l_i, l_j\}$. Then, as $u_k \geq l_k$ and $u_k \geq max_{s\neq i,j,k}\{u_p\}$, we have $u_k \geq max_{p\neq i,j}\{l_p\}$.
And we have $l_{max} = max\{l_1, \cdots, l_n\} = max\{l_i, l_j, u_k\} = max\{l_i, l_j\}$.
If $l_{max} = l_i$, then we have $l_i < u_j \wedge (l_i < u_k \vee l_i \geq u_j)$, that is, $l_i < u_j \wedge l_i < u_k$. It contradicts $u_k \leq max\{l_i, l_j\}$.
if $l_{max} = l_j \wedge l_{max} \neq l_i$, then we have $l_j > u_j \vee l_j < u_k$, that is $l_j < u_k$. It contradicts $u_k \leq max\{l_i, l_j\}$.
Therefore, in case 4, $u_k > max\{l_i, l_j\}$.
Here, we prove the upper bound in case 4 is sound.
$f(x_1, \cdots, x_n) = max(x_1, \cdots, x_n), \forall(x_1, \cdots, x_n)$. If $f(x_1, \cdots, x_n) = x_i$,

$$\begin{aligned}
u(x_1, \cdots, x_n) - x_i =& \frac{u_i - u_k}{u_i - l_i}(x_i - l_i) + \frac{u_j - u_k}{u_j - l_j}(x_j - l_j) + u_k - x_i \\
=& \frac{u_i - u_k}{u_i - u_k}(x_i - l_i) + \frac{u_j - u_k}{u_j - l_j}(x_j - l_j) + u_k - l_i + l_i - x_i \\
=& (x_i - l_i)(\frac{u_i - u_k}{u_i - l_i} - 1) + \frac{u_j - u_k}{u_j - l_j}(x_j - l_j) + u_k - l_i \\
=& (u_k - l_i)\frac{u_i - x_i}{u_i - l_i} + \frac{u_j - u_k}{u_j - l_j}(x_j - l_j) \\
\geq& 0
\end{aligned}$$

If $f(x_1, \cdots, x_n) = x_j$, the proof is the same as above.

$$\begin{aligned}
u(x_1, \cdots, x_n) - x_j =& \frac{u_i - u_k}{u_i - l_i}(x_i - l_i) + \frac{u_j - u_k}{u_j - l_j}(x_j - l_j) + u_k - x_j \\
=& \frac{u_i - u_k}{u_i - u_k}(x_i - l_i) + \frac{u_j - u_k}{u_j - l_j}(x_j - l_j) + u_k - l_j + l_j - x_j \\
=& \frac{u_i - u_k}{u_i - l_i}(x_i - l_i) + (\frac{u_j - u_k}{u_j - l_j} - 1)(x_j - l_j) + u_k - l_j \\
=& \frac{u_i - u_k}{u_i - l_i}(x_i - l_i) + (1 - \frac{x_j - l_j}{u_j - l_j})(u_k - l_j) \\
=& \frac{u_i - u_k}{u_i - l_i}(x_i - l_i) + \frac{u_j - x_j}{u_j - l_j}(u_k - l_j) \\
\geq& 0
\end{aligned}$$

If $f(x_1, \cdots, x_n) = x_k$,

$$\begin{aligned}
u(x_1, \cdots, x_n) - x_j =& \frac{u_i - u_k}{u_i - l_i}(x_i - l_i) + \frac{u_j - u_k}{u_j - l_j}(x_j - l_j) + (u_k - x_k) \\
\geq& 0
\end{aligned}$$

If $f(x_1, \cdots, x_n) = x_l, l \neq i, j, k$,

$$\begin{aligned}
& u(x_1, \cdots, x_n) - x_l \\
=& \frac{u_i - u_k}{u_i - l_i}(x_i - l_i) + \frac{u_j - u_k}{u_j - l_j}(x_j - l_j) + u_k - x_l \\
=& \frac{u_i - u_k}{u_i - l_i}(x_i - l_i) + \frac{u_j - u_k}{u_j - l_j}(x_j - l_j) + (u_k - u_l) + (u_l - x_l) \\
\geq& 0
\end{aligned}$$

Then, we have $u(\boldsymbol{x}) \geq f(\boldsymbol{x})$, that is, the upper linear bound of case 4 is sound.
Lower linear bound:

$l(x_1, \cdots, x_n) = x_j = argmax_i m_i$, and $\forall (x_1, \cdots, x_n) \in \times_{i=1}^n [l_i, u_i]$,

$$\begin{aligned} f(x_1, \cdots, x_n) &= max(x_1, \cdots, x_n) \\ &\geq x_j \\ &= l(x_1, \cdots, x_n) \end{aligned}$$

Then, we have $l(\boldsymbol{x}) \leq f(\boldsymbol{x})$, that is the lower linear bound is sound.

This completes the proof. □

7.2. Proof of Theorem 2

In this subsection, we prove Theorem 2.

Proof. First, $u(\boldsymbol{x})$ and $l(\boldsymbol{x})$ are upper bound and lower bound for $f(\boldsymbol{x})$. Therefore, $u(\boldsymbol{x}) \geq l(\boldsymbol{x}), \forall \boldsymbol{x} \in [\boldsymbol{l}, \boldsymbol{u}]$. Then, we have

$$u(\boldsymbol{x}) - l(\boldsymbol{x}) = (u(\boldsymbol{x}) - f(\boldsymbol{x})) + (f(\boldsymbol{x}) - l(\boldsymbol{x}))$$

$(u(\boldsymbol{x}) - f(\boldsymbol{x}))$ and $(f(\boldsymbol{x}) - l(\boldsymbol{x}))$ are not smaller than 0, when $\forall \boldsymbol{x} \in [\boldsymbol{l}, \boldsymbol{u}]$.

Therefore, minimizing $\iint_{[\boldsymbol{l},\boldsymbol{u}]}(u(\boldsymbol{x}) - l(\boldsymbol{x}))d\boldsymbol{x}$ is equivalent to minimizing both $\iint_{[\boldsymbol{l},\boldsymbol{u}]}(u(\boldsymbol{x}) - f(\boldsymbol{x}))d\boldsymbol{x}$ and $\iint_{[\boldsymbol{l},\boldsymbol{u}]}(f(\boldsymbol{x}) - l(\boldsymbol{x}))d\boldsymbol{x}$. The value of $\iint_{[\boldsymbol{l},\boldsymbol{u}]} f(\boldsymbol{x})d\boldsymbol{x}$ is constant. Thus, it is also equivalent to minimizing $\iint_{[\boldsymbol{l},\boldsymbol{u}]} u(\boldsymbol{x})d\boldsymbol{x}$ and $\iint_{[\boldsymbol{l},\boldsymbol{u}]}(-l(\boldsymbol{x}))d\boldsymbol{x}$. Further, we define that lower linear bound $l(\cdot)$ is the neuron-wise tightest when $-l(\boldsymbol{m})$ reaches the minimum, and upper linear bound $u(\cdot)$ is the neuron-wise tightest when $u(\boldsymbol{m})$ reaches the minimum.

Define $\boldsymbol{x} = (x_1, \cdots, x_n)$ and $[n] = \{1, 2, \cdots, n\}$. Because $u(\boldsymbol{x})$ is a linear combination of $x_i, i \in [n]$. Without loss of generality, we assume $u(\boldsymbol{x}) = \sum_{i \in [n]} a_{u,i} x_i + b_u$. Then,

$$\begin{aligned} \iint_{(x_1, \cdots, x_n) \in [\boldsymbol{l},\boldsymbol{u}]} u(x_1, \cdots, x_n) d\boldsymbol{x} &= \iint_{(x_1, \cdots, x_n) \in [\boldsymbol{l},\boldsymbol{u}]} (\sum_{i \in [n]} a_{u,i} x_i + b_u) d\boldsymbol{x} \\ &= \sum_{i=1}^n \frac{a_{u,i}}{2}((u_i)^2 - (l_i)^2) + b_u(u_i - l_i) \\ &= \prod_{i=1}^n (u_i - l_i)(\sum_{i \in [n]} a_{u,i} \frac{u_i + l_i}{2} + b_u) \\ &= \prod_{i=1}^n (u_i - l_i) u(\boldsymbol{m}) \end{aligned}$$

where $\boldsymbol{m} = (\frac{u_1 + l_1}{2}, \cdots, \frac{u_n + l_n}{2})$. As $u_i, l_i, i \in [n]$ are constant, the minimize target has been transformed into minimizing $u(\boldsymbol{m})$.

Symmetric to the above proof, minimizing $\iint_{(x_1, \cdots, x_n) \in [\boldsymbol{l},\boldsymbol{u}]} u(x_1, \cdots, x_n) d\boldsymbol{x}$ is equivalent to minimize $-l(\boldsymbol{m})$.

This completes the proof. □

7.3. Proof of Theorem 3

In this subsection, we prove Theorem 3, that is, our linear bounds in Theorem 1 are the neuron-wise tightest.

Proof. Define $\boldsymbol{m} = \frac{\boldsymbol{u} + \boldsymbol{l}}{2} = (m_1, \cdots, m_n) \in \mathbb{R}^n$.

Upper linear bound:

Case 1: When $l_i = l_{max} \wedge l_{max} \geq u_j$, we have $f(x_1, \cdots, x_n) = x_i$ and $u(x_1, \cdots, x_n) = x_i - l_i + l_i = x_i$. As $u(m) = f(m) \leq u'(\boldsymbol{m}), \forall u' \in \mathcal{U}$, the upper bound is the neuron-wise tightest.

Case 2: When $l_i = l_{max}, u_j > l_i \geq u_k$, we have $f(x_1, \cdots, x_n) = max(x_i, x_j)$ and $u(x_1, \cdots, x_n) = x_i - l_i + \frac{u_j - l_i}{u_j - l_j}(x_j - l_j) + l_i$.

Because

$$\begin{aligned}
&u(u_1, \cdots, u_{i-1}, l_i, u_{i+1}, \cdots, u_{j-1}, u_j, u_{j+1}, \cdots, u_n) \\
=&l_i - l_i + \frac{u_j - l_i}{u_j - l_j}(u_j - l_j) + l_i \\
=&u_j \\
=&f(u_1, \cdots, u_{i-1}, l_i, u_{i+1}, \cdots, u_{j-1}, u_j, u_{j+1}, \cdots, u_n)
\end{aligned}$$

and

$$\begin{aligned}
&u(l_1, \cdots, l_{i-1}, u_i, l_{i+1}, \cdots, l_{j-1}, l_j, l_{j+1}, \cdots, l_n) \\
=&u_i - l_i + \frac{u_j - l_i}{u_j - l_j}(l_j - l_j) + l_i \\
=&u_i \\
=&f(l_1, \cdots, l_{i-1}, u_i, l_{i+1}, \cdots, l_{j-1}, l_j, l_{j+1}, \cdots, l_n)
\end{aligned}$$

We notice that
$\boldsymbol{a} := (u_1, \cdots, u_{i-1}, l_i, u_{i+1}, \cdots, u_n)$ and $\boldsymbol{b} := (l_1, \cdots, l_{i-1}, u_i, l_{i+1}, \cdots, l_n)$ are the space diagonal of $\times_{i=1}^{n}[l_i, u_i]$, and $\boldsymbol{m} = \frac{1}{2}(\boldsymbol{a} + \boldsymbol{b})$. Then, as $u(\boldsymbol{x})$ is linear, we have $f(\boldsymbol{a}) + f(\boldsymbol{b}) = f(\boldsymbol{a} + \boldsymbol{b})$.
Then, we have

$$\begin{aligned}
\forall u' \in \mathcal{U}, u(\boldsymbol{m}) &= u(\frac{1}{2}(\boldsymbol{a} + \boldsymbol{b})) \\
&= \frac{1}{2}(u(\boldsymbol{a}) + u(\boldsymbol{b})) \\
&= \frac{1}{2}(f(\boldsymbol{a}) + f(\boldsymbol{b})) \\
&\leq \frac{1}{2}(u'(\boldsymbol{a}) + u'(\boldsymbol{b})) \\
&= (u'(\frac{1}{2}(\boldsymbol{a} + \boldsymbol{b}))) \\
&= u'(\boldsymbol{m})
\end{aligned}$$

Therefore, the plane is the neuron-wise tightest upper linear bounding plane.

Case 3: When $l_i = l_{max} \wedge l_i \neq l_{max} \wedge u_j \geq l_j \geq u_k$, we have $f(x_1, \cdots, x_n) = max(x_i, x_j)$ and $u(x_1, \cdots, x_n) = \frac{u_i - l_j}{u_i - l_i}x_i + x_j + l_j$.
Because

$$\begin{aligned}
&u(u_1, \cdots, u_{i-1}, u_i, u_{i+1}, \cdots, u_{j-1}, l_j, u_{j+1}, \cdots, u_n) \\
=&\frac{u_i - l_j}{u_i - l_i}(u_i - l_i) + (l_j - l_j) + l_j \\
=&f(u_1, \cdots, u_{i-1}, u_i, u_{i+1}, \cdots, u_{j-1}, l_j, u_{j+1}, \cdots, u_n)
\end{aligned}$$

and

$$\begin{aligned}
&u(l_1, \cdots, l_{i-1}, l_i, l_{i+1}, \cdots, l_{j-1}, u_j, l_{j+1}, \cdots, l_n) \\
=&\frac{u_i - l_j}{u_i - l_i}(l_i - l_i) + (u_j - l_j) + l_j \\
=&f(l_1, \cdots, l_{i-1}, l_i, l_{i+1}, \cdots, l_{j-1}, u_j, l_{j+1}, \cdots, l_n)
\end{aligned}$$

We notice that
$\boldsymbol{a} := (u_1, \cdots, u_{j-1}, l_j, u_{j+1}, \cdots, u_n)$ and $\boldsymbol{b} := (l_1, \cdots, l_{j-1}, u_j, l_{j+1}, \cdots, l_n)$ are the space diagonal of $\times_{i=1}^{n}[l_i, u_i]$, and
$\boldsymbol{m} = \frac{1}{2}(\boldsymbol{a}+\boldsymbol{b})$. Similar to the proof in case 2, we can prove that the plane is the neuron-wise tightest linear upper bounding plane.

Case 4: First, as proved in Section 7.1, $u_k > max\{l_i, l_j\}$ in case 4.

Because

$$\begin{aligned}
&u(u_1, \cdots, u_{i-1}, u_i, u_{i+1}, \cdots, u_{j-1}, l_j, u_{j+1}, \cdots, u_n) \\
=&\frac{u_i - u_k}{u_i - l_i}(u_i - l_i) + \frac{u_j - u_k}{u_j - l_j}(l_j - l_j) + u_k \\
=&u_i - u_k + u_k \\
=&f(u_1, \cdots, u_{i-1}, u_i, u_{i+1}, \cdots, u_{j-1}, l_j, u_{j+1}, \cdots, u_n)
\end{aligned}$$

and

$$\begin{aligned}
&u(l_1, \cdots, l_{i-1}, l_i, l_{i+1}, \cdots, l_{j-1}, u_j, l_{j+1}, \cdots, l_n) \\
=&\frac{u_i - u_k}{u_i - l_i}(l_i - l_i) + \frac{u_j - u_k}{u_j - l_j}(u_j - l_j) + u_k \\
=&u_j - u_k + u_k \\
=&f(l_1, \cdots, l_{i-1}, l_i, l_{i+1}, \cdots, l_{j-1}, u_j, l_{j+1}, \cdots, l_n)
\end{aligned}$$

As

$$\begin{aligned}
\boldsymbol{m} =&\frac{1}{2}(u_1, \cdots, u_{i-1}, u_i, u_{i+1}, \cdots, u_{j-1}, l_j, u_{j+1}, \cdots, u_n) \\
&+\frac{1}{2}(l_1, \cdots, l_{i-1}, l_i, l_{i+1}, \cdots, l_{j-1}, u_j, l_{j+1}, \cdots, l_n)
\end{aligned}$$

We notice that

$\boldsymbol{a} := (l_1, \cdots, l_{j-1}, u_j, l_{j+1}, \cdots, l_n)$ and $\boldsymbol{b} := (u_1, \cdots, u_{j-1}, l_j, u_{j+1}, \cdots, u_n))$ are the space diagonal of $\times_{i=1}^{n}[l_i, u_i]$, and $\boldsymbol{m} = \frac{1}{2}(\boldsymbol{a} + \boldsymbol{b})$. Similar to the proof in case 2, we can prove that the plane is the neuron-wise tightest linear upper bounding plane.

Lower linear bound:

$l(x_1, \cdots, x_n) = x_j = argmax_i m_i$, and $\forall (x_1, \cdots, x_n) \in \times_{i=1}^{n}[l_i, u_i]$,

$l(\boldsymbol{m}) = f(m_1, \cdots, m_n) \geq l'(\boldsymbol{m}), \forall l \in \mathcal{L}$, hence, $l(x_1, \cdots, x_n)$ is the neuron-wise tightest lower bounding plane.

This completes the proof. □

Table 4. The additional experimental setup and source of neural networks used in experiments.

Dataset	Network	#Nodes	Accuracy	#Properties	ϵ	Source
MNIST	Conv_MaxPool	14592	81.9	81	2/255	ERAN
	Convnet_MaxPool	25274	98.8	96	10/255	Verivital
	CNN, 4 layers	36584	99.0	94	3/255	CNN-Cert
	CNN, 5 layers	52872	99.1	99	2/255	
	CNN, 6 layers	56392	99.1	99	2/255	
	CNN, 7 layers	56592	99.1	91	2/255	
	CNN, 8 layers	56912	99.3	99	2/255	
CIFAR-10	Conv_MaxPool	57020	44.6	48	0.001	ERAN
	CNN, 4 layers	49320	71.3	27	0.001	CNN-Cert
	CNN, 5 layers	71880	71.1	16	0.001	
	CNN, 6 layers	77576	73.8	14	0.001	
	CNN, 7 layers	77776	75.6	29	0.001	
	CNN, 8 layers	78416	68.1	12	0.001	
ModelNet40	16p_Natural	26200	76.8	59	0.005	3DCertify
	32p_Natural	49800	83.2	62	0.005	
	64p_Natural	97000	85.7	64	0.005	
	64p_FGSM	97000	86.0	66	0.01	
	64p_IBP	97000	78.1	60	0.01	

Table 5. The performance of Ti-Lin on CNN-Cert (backsubstitution-based).

Dataset	Network	l_p	Certified Bounds(10^{-5}) CNN-Cert	Certified Bounds(10^{-5}) Ti-Lin, on CNN-Cert	Bound Impr.(%) vs. CNN-Cert	Average Runtime(min) CNN-Cert	Average Runtime(min) Ti-Lin, on CNN-Cert
MNIST	CNN	l_∞	1318	**1837**	39.4	1.8	1.7
	4 layers	l_2	4427	**6478**	46.3	1.4	1.4
	36584 nodes	l_1	8544	**12642**	48.0	1.4	1.4
	CNN	l_∞	1288	**1817**	41.0	8.4	8.8
	5 layers	l_2	5164	**7359**	42.5	11.9	9.2
	52872 nodes	l_1	10147	**14292**	40.9	10.8	9.5
	CNN	l_∞	1025	**1382**	34.8	20.5	20.9
	6 layers	l_2	3954	**5409**	36.8	20.6	20.4
	56392 nodes	l_1	7708	**10455**	35.6	20.6	20.0
	CNN	l_∞	647	**930**	43.7	24.7	24.6
	7 layers	l_2	2733	**4022**	47.2	25.1	23.8
	56592 nodes	l_1	5443	**8002**	47.0	22.9	22.9
	CNN	l_∞	847	**1221**	44.2	26.5	26.7
	8 layers	l_2	3751	**5320**	41.8	25.0	24.9
	56912 nodes	l_1	7515	**10655**	41.8	23.7	24.2
	LeNet_ReLU	l_∞	1204	**1864**	54.8	0.2	0.2
	3 layers	l_2	6534	**10862**	66.2	0.2	0.2
	8080 nodes	l_1	17937	**30305**	69.0	0.2	0.2
	LeNet_Sigmoid	l_∞	1684	**2042**	21.3	0.3	0.3
	3 layers	l_2	9926	**12369**	24.6	0.3	0.3
	8080 nodes	l_1	26937	**33384**	23.9	0.3	0.3
	LeNet_Tanh	l_∞	613	**817**	33.3	0.3	0.3
	3 layers	l_2	3462	**4916**	42.0	0.3	0.3
	8080 nodes	l_1	9566	**13672**	42.9	0.3	0.3
	LeNet_Atan	l_∞	617	**836**	35.5	0.3	0.3
	3 layers	l_2	3514	**5010**	42.6	0.3	0.3
	8080 nodes	l_1	9330	**13345**	43.0	0.3	0.3
CIFAR-10	CNN	l_∞	108	**129**	19.4	3.1	2.9
	4 layers	l_2	751	**1038**	38.2	2.5	2.5
	49320 nodes	l_1	2127	**3029**	42.4	2.5	2.5
	CNN	l_∞	115	**146**	27.0	13.1	13.0
	5 layers	l_2	953	**1342**	40.8	12.4	12.7
	71880 nodes	l_1	2850	**4087**	43.4	12.3	12.6
	CNN	l_∞	99	**120**	21.2	28.6	28.6
	6 layers	l_2	830	**1078**	29.9	27.6	27.9
	77576 nodes	l_1	2387	**3174**	33.0	27.7	27.4
	CNN	l_∞	66	**83**	25.8	33.4	33.3
	7 layers	l_2	573	**773**	34.9	32.5	32.8
	77776 nodes	l_1	1673	**2303**	37.7	33.6	32.6
	CNN	l_∞	56	**70**	25.0	36.9	37.5
	8 layers	l_2	536	**705**	31.5	37.5	36.6
	78416 nodes	l_1	1609	**2160**	34.2	36.9	37.0
Tiny ImageNet	CNN	l_∞	77	**123**	59.7	184.9	184.0
	7 layers	l_2	580	**939**	61.9	184.4	183.3
	703512 nodes	l_1	1747	**2875**	64.6	193.6	183.9

7.4. Experiment setup

In this subsection, we present some experiment setups of Section 5 in detail. Concretely, we list the number of nodes, the sources of networks, the number of Properties to be verified, and the perturbation range ϵ in Table 4. Following the setting of ERAN, we generate properties for all networks by the correctly classified inputs in the first 100 inputs. We evaluate our method on four datasets, including MNIST, a dataset of 28×28 handwritten digital images in 10 classes, CIFAR-10, a dataset

Table 6. The performance of Ti-Lin and MaxPool2ReLU on α, β-CROWN verifier.

		Certified accuracy(%)		Avg. time of safe instances(s)		Speedup
Network	Radius	MaxPool2ReLU	Ti-Lin	MaxPool2ReLU	Ti-Lin	vs. MaxPool2ReLU
MNIST_Conv_MaxPool	0.008	56.8	**65.4**	32.5	**17.3**	1.9
	0.009	49.4	**60.5**	46.3	**25.2**	1.8
	0.010	40.7	**54.3**	65.9	**32.5**	2.0
ConvNet_MaxPool	0.020	60.0	**80.0**	191.0	**0.6**	318.3
	0.030	15.0	**50.0**	377.5	**37.5**	10.1
	0.040	0.0	**20.0**	-	**44.2**	-
		Timeout rate(%)		**Avg. time of all instances(s)**		**Speedup**
Network	Radius	MaxPool2ReLU	Ti-Lin	MaxPool2ReLU	Ti-Lin	vs. MaxPool2ReLU
MNIST_Conv_MaxPool	0.008	37.0	**28.4**	494.9	**65.5**	7.6
	0.009	43.2	**32.1**	667.8	**77.4**	8.6
	0.010	50.6	**37.0**	770.1	**88.7**	8.7
ConvNet_MaxPool	0.020	20.0	**0.0**	294.6	**0.7**	420.9
	0.030	60.0	**20.0**	549.5	**104.7**	5.2
	0.040	45.0	**5.0**	566.1	**47.5**	11.9

Table 7. The performance of Ti-Lin and MaxPool2ReLU on ERAN framework.

		Certified accuracy(%)		Averaged Time(s)		Speedup
Network	Radius	MaxPool2ReLU	Ti-Lin	MaxPool2ReLU	Ti-Lin	vs. MaxPool2ReLU
MNIST_Conv_MaxPool	0.006	24.7	**69.1**	645.3	**18.5**	34.9
	0.007	11.1	**64.2**	776.2	**25.7**	30.2
	0.008	7.4	**45.7**	882.3	**36.9**	23.9
ConvNet_MaxPool	0.020	0.0	**65.0**	984.3	**86.5**	11.4
	0.030	0.0	**50.0**	1022.5	**153.5**	6.7
	0.040	0.0	**35.0**	973.6	**149.6**	6.5

of 60,000 $32 \times 32 \times 3$ images in 10 classes, e.g., airplane, bird, and ship, Tiny ImageNet [13], a dataset of 100,000 $64 \times 64 \times 3$ images in 200 classes, and ModelNet40, a dataset of 12,311 pre-aligned shapes from 40 categories.

7.5. Additional experiments

In this subsection, we conduct some additional experiments to further illustrate (I) the performance of Ti-Lin on CNN-Cert, a back-substitution-based verifier. (II) We compare Ti-Lin to OSIP to illustrate the superiority of tight linear approximation for MaxPool over the MaxPool2ReLU transformation.

7.5.1. Results (I): performance on CNN-Cert

To compare Ti-Lin to CNN-Cert fairly, we implement Ti-Lin on CNN-Cert. We follow the metrics used in CNN-Cert. We use the certified robustness bound introduced in Section 3 as the tightness metric and the average computation time as the efficiency metric. As for the improvement of tightness, we use $\frac{100(\epsilon_l'-\epsilon_l)}{\epsilon_l}\%$ to quantify the percentage of improvement, where ϵ_l' and ϵ_l represent the average certified lower bounds certified by Ti-Lin and CNN-Cert, respectively. We evaluate Ti-Lin and CNN0Cert on 10 inputs for all CNNs in Table 5. The initial perturbation range is 0.005. Testing on 10 inputs can sufficiently evaluate the performance of verification methods, as it is shown that the average certified results of 1000 inputs are similar to 10 images [5]. The results are shown in Table 5. The results indicates that, under different networks, datasets, perturbation norm(l_1, l_2, l_∞), Ti-Lin's certified bounds achieved better performance than CNN-Cert without incurring additional time overhead. Specifically, Ti-Lin exhibits larger robustness bounds compared to CNN-Cert with improvements of up to 69.0, 43.3, 64.6% on the MNIST, CIFAR-10, and Tiny ImageNet datasets, respectively.

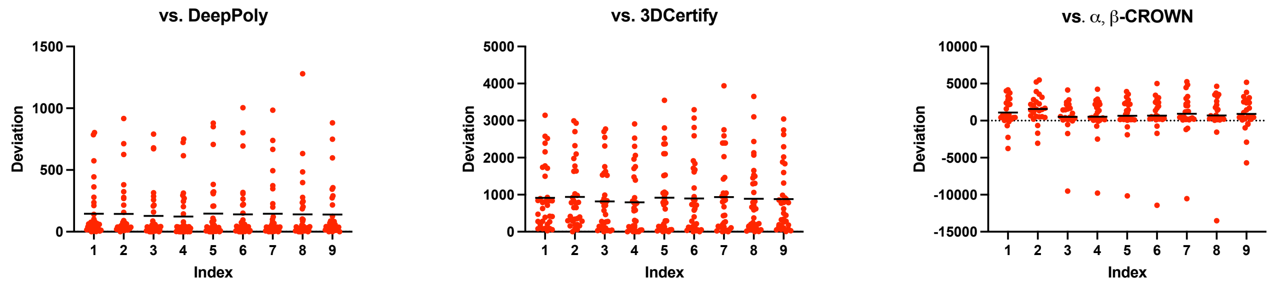

Figure 3. Visualization of the global lower bounds verified by DeepPoly, 3DCertify, α, β-CROWN, and Ti-Lin. Red dots represent the deviation of the global bounds $\boldsymbol{L} - \boldsymbol{L}'$, where $\boldsymbol{L}$ and $\boldsymbol{L}'$ represent the global bounds of Ti-Lin and other methods testing on 100 inputs, respectively. Black lines represent the mean of the deviations.

7.5.2. Results (II): comparison to MaxPool2ReLU transformation method

Notably, as OSIP [26] is not open-sourced, we use another alternative method, called MaxPool2ReLU for evaluation. Concretely, we follow OSIP to transform the MaxPool-based network into a ReLU-based network. ERAN, employing advanced multi-ReLU relaxation for ReLU, and α, β-CROWN, leveraging the branch-and-bound technique for ReLU neurons, are two cutting-edge verifiers designed for handling the ReLU layer. Therefore, we use the results of the transformed networks tested by ERAN and α, β-CROWN as the alternative results of OSIP, denoted as MaxPool2ReLU. The results are shown in Table 6 and Table 7. The results show that MaxPool2ReLU transformation would lead to coarse certified results and much more time consumption. Concretely, Ti-Lin computes higher certified accuracy with up to 35% and 65% improvement than MaxPool2ReLU on α, β-CROWN and ERAN, respectively. Further, Ti-Lin can accelerate the verification process with up to 420.9$\times$ and 34.9$\times$ speedup regarding the average time of all instances on α, β-CROWN and ERAN, respectively. This is because the transformation would make the network deeper, leading to the cumulative overapproximation region and much time consumption to verify. For example, when the pool size is 2×2, one MaxPool layer can be transformed into three ReLU layers and three affine layers. Thus, Ti-Lin is much faster and tighter than MaxPool2ReLU. Especially when verifying ConvNet_MaxPool, the verified accuracy of MaxPool2ReLU (ERAN) is zero across all perturbation radii, while Ti-Lin can at least verify 35% inputs when the perturbation radius is 0.040, respectively.

7.5.3. Result (III): Evaluation on global lower bounds

According to Equation 2, we decide whether the perturbed region is safe based on $l_t^K - u_j^K \geq 0, j \neq t, j \in [n_K]$. Therefore, the global lower bounds $\boldsymbol{L} := (l_t^K - u_j^K), j \neq t, j \in [n_K]$ is the raw criterion to evaluate the tightness. To further illustrate the advantages of the neuron-wise tightness over other linear bounds, we analyze the global lower bounds of the last layer computed by Ti-Lin and other methods, including DeepPoly, 3DCertify, and optimized linear bounds, used in α, β-CROWN. In Figure 3, we show the deviation of the global lower bounds of Ti-Lin and the baseline on CIFAR_Conv_MaxPool. The x-axis represents the index of the global lower bound (labels without the true label), and the y-axis represents the deviations between the global lower bounds. As shown in Figure 3, Ti-Lin has larger global bounds on all inputs than DeepPoly and 3DCertify and on most inputs than α, β-CROWN. Further, the mean of the deviations $\boldsymbol{L} - \boldsymbol{L}'$ are all larger than zero. It reveals that the neuron-wise tightest linear bounds can bring tighter output bounds than other methods. Consequently, Ti-Lin can certify much larger certified accuracy than these methods in Table 1 and 3.

According to the BaB design of α, β-CROWN, we compare the global lower bound for verifying the property 0(true label agaist label 0) on ConvNet_MaxPool, with the results illustrated in Figure 4. Initially, at BaB round = 0, MaxLin achieves a higher global lower bound compared to Ti-Lin. This is attributed to MaxLin's block-wise tightest approach, which is specifically designed for ReLU + MaxPool blocks, giving it an early advantage in representing tighter bounds before the BaB analysis progresses. However, as the BaB process progresses, the α, β-CROWN verification framework employs plane-cutting techniques for ReLU neurons. This enables Ti-Lin's neuron-wise tightest method to leverage its finer granularity, which significantly improves the global lower bound in later BaB rounds. Consequently, Ti-Lin surpasses MaxLin, achieving a much higher global lower bound in the later stages of analysis. This comparison demonstrates that while MaxLin provides better bounds in the initial stages, Ti-Lin's more precise neuron-wise tightness proves to be advantageous for verifying robustness after iterative BaB analysis.

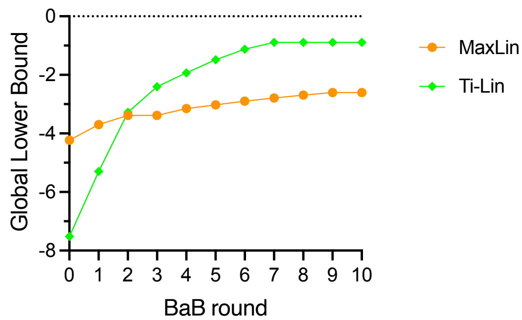

Figure 4. Visualization of the global lower bound verified by MaxLin and Ti-Lin, both of which are built upon the framework of α, β-CROWN.

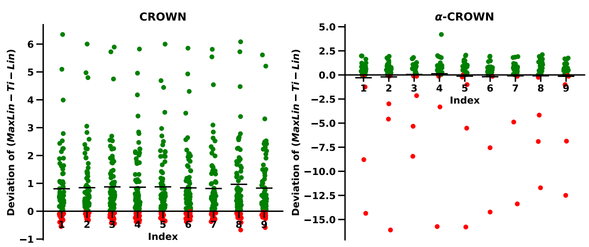

Figure 5. The deviation in global bounds between MaxLin and Ti-Lin when using the CROWN and α-CROWN frameworks.

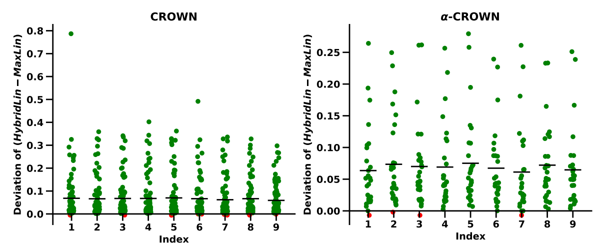

Figure 6. The deviation in global bounds between Hybrid-Lin and MaxLin when using the CROWN and α-CROWN frameworks.

7.5.4. Result (IV): Hybrid-Lin: a combination of Ti-Lin and MaxLin

MaxLin and Ti-Lin make distinct but non-contradictory claims. MaxLin provides the **block-wise tightest** upper linear bound when the ReLU's upper linear bound is $u(x) = \frac{u}{u-l}(x - l)$, while Ti-Lin, being **neuron-wise tightest**, also achieves block-wise tightest bounds when the ReLU's upper linear bound is $u(x) = 0$ or $u(x) = x$. Thus, Ti-Lin outperforms MaxLin when ReLU's upper bound incurs no precision loss. Rather than competing, they complement each other in achieving optimal bounding precision. To illustrate this, we introduce HybridLin, which uses MaxLin when one of the ReLU' upper linear bound is $u(x) = \frac{u}{u-l}(x - l)$ while Ti-Lin when all the ReLU's upper linear bound is $u(x) = 0$ or $u(x) = x$. In Figures 5 and 6, green points indicate positive deviation, red indicate negative. Figure 5 shows that when ReLU's upper linear bound incurring precision loss, MaxLin's performance varies, underscoring the importance of Ti-Lin in achieving tighter results. Figure 6 shows that Hybrid-Lin, using Ti-Lin when ReLU's upper linear bound is precise, consistently achieves tighter bounds than MaxLin.