Supplementary Document

3DVerifier: Efficient Robustness Verification for 3D Point Cloud Models

A The Power of JANet

We perform ablation study to show the importance of JANet in the point clouds classification task. We choose three models to train, and record the number of trainable parameters for each model. The final results of training accuracy are demonstrated in Table 1. The specific layer configurations for these models are presented in Appendix C. As the PointNet without JANet can be regarded as a general CNN model, while JANet is a complex architecture that contains T-Net and multiplication layer, to interpret the effect of JANet, we examine two models for the PointNet without JANet: 1) 7-layer model, which abandons the JANet in the 13-layer full PointNet; 2) 13-layer model, which intuitively adds the convolution and dense layers in T-Net to the 7-layer model. From the results, we can see that the PointNet with JANet can improve the training accuracy significantly comparing with PointNet without JANet when they employ the same number of layers. Therefore, JANet plays an important role to improve the performance of point clouds models.

B Configuration of PointNet models

Below are the specific layer configurations for 13-layer full PointNet with JANet and PointNets without JANet.

Table 1: Training accuracy of different structures of point cloud models.

Models	N	no. trainable parameters	accuracy
PointNet without JANet	7	443656	72.94%
PointNet without JANet	13	822472	74.07%
Full PointNet with JANet	13	645425	81.31%

No	Type	no. features	normalization	activation function
1	Conv1D	32	BatchNormalization	ReLU
2	Conv1D	64	BatchNormalization	ReLU
3	Conv1D	512	BatchNormalization	ReLU
4	GlobalMaxpooling			
5	Dense	256	BatchNormalization	ReLU
6	Dense	512	BatchNormalization	ReLU
	Reshape			
	Multiplication			

Table 2: Configuration for T-Net.

No	Type	no. features	normalization	activation function
	T-Net			
7	Conv1D	32	BatchNormalization	ReLU
8	Conv1D	64	BatchNormalization	ReLU
9	Conv1D	512	BatchNormalization	ReLU
10	GlobalMaxpooling			
11	Dense	512	BatchNormalization	ReLU
12	Dense	256	BatchNormalization	ReLU
	Dropout			

Table 3: Configuration for full PointNet with JANet.

No	Type	no. features	normalization	activation function
1	Conv1D	32	BatchNormalization	ReLU
2	Conv1D	32	BatchNormalization	ReLU
3	Conv1D	64	BatchNormalization	ReLU
4	Conv1D	512	BatchNormalization	ReLU
5	GlobalMaxpooling			
6	Dense	512	BatchNormalization	ReLU
7	Dense	256	BatchNormalization	ReLU
	Dropout			

Table 4: Configuration for 7-layer PointNet without JANet.

No	Type	no. features	normalization	activation function
1	Conv1D	32	BatchNormalization	ReLU
2	Conv1D	32	BatchNormalization	ReLU
3	Conv1D	64	BatchNormalization	ReLU
4	Conv1D	64	BatchNormalization	ReLU
5	Conv1D	512	BatchNormalization	ReLU
6	Conv1D	512	BatchNormalization	ReLU
7	GlobalMaxpooling			
8	Dense	512	BatchNormalization	ReLU
9	Dense	256	BatchNormalization	ReLU
10	Dense	256	BatchNormalization	ReLU
11	Dense	128	BatchNormalization	ReLU
12	Dense	128	BatchNormalization	ReLU
	Dropout			

Table 5: Configuration for 12-layer PointNet without JANet.

C Derivation of the linear functions to bound the multiplication layer

The mathematical proof to choose the optimal parameters of the linear functions to bound the multiplication layer is evaluated in Shi et al (2020). Here we will

present how to derive the optimisation problem based on the derivation in Shi et al (2020).

The goal of the linear relaxation is to bound the multiplication on two inputs, $p_1 p_2$, which can be represented by the following constraints:

$$a^L p_1 + b^L p_2 + c^L \leq p_1 p_2 \leq a^U p_1 + b^U p_2 + c^U \tag{1}$$

As we have already obtained the bounds of p_1 and p_2, given the bounds of p_1 is $[l_1, u_1]$ and for p_2 is $[l_2, u_2]$, we can choose the appropriate parameters of a^L, a^U, b^L, b^U, c^L and c^U according to the computed bounds.

C.1 derivation for the lower bound

Let the objective function be :

$$G^L(p_1, p_2) = p_1 p_2 - (a^L p_1 + b^L p_2 + c^L) \tag{2}$$

Thus, to find the appropriate parameters, we need find the optimal value of a^L, b^L and c^L to minimize the gap function G, $minG^L(p_1, p_2)$. As the p_1 and p_2 are constrained by the area $[l_1, u_1] \times [l_2, u_2]$. Thus, we formulate the objective function as :

$$\max_{a^L, b^L, c^L} F^L(p_1, p_2) = \max_{a^L, b^L, c^L} \int_{p_1 \in [l_1, u_1]} \int_{p_2 \in [l_2, u_2]} a^L p_1 + b^L p_2 + c^L \tag{3}$$

s.t. $G^L(p_1, p_2) \geq 0$ $(\forall (p_1, p_2) \in [l_1, u_1] \times [l_2, u_2])$. First, we need to ensure that $G^L(p_1, p_2) \geq 0$ by finding the minimum value of $G(p_1, p_2)$. The first-order partial derivatives of G^L are:

$$\begin{array}{l} \frac{\partial G^L}{\partial p_1} = p_2 - a^L \\ \frac{\partial G^L}{\partial p_2} = p_1 - b^L \end{array} \tag{4}$$

The critical points are $\frac{\partial G^L}{\partial p_1} = 0$ and $\frac{\partial G^L}{\partial p_2} = 0$, which leads to $p_2 = a^L$ and $p_1 = b^L$. Then we form the Hessian matrix to justify the behavior of the critical points:

$$\mathrm{H}(G^L(p_1, p_2)) = \begin{pmatrix} \frac{\partial^2 G^L}{\partial p_1^2} & \frac{\partial^2 G^L}{\partial p_1 \partial p_2} \\ \frac{\partial^2 G^L}{\partial p_2 \partial p_1} & \frac{\partial^2 G^L}{\partial p_2^2} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{5}$$

Then we compute the Hessian:

$$\det \mathrm{H}(G^L(p_1, p_2)) = \left(\frac{\partial^2 G^L}{\partial p_1^2}\right)\left(\frac{\partial^2 G^L}{\partial p_2^2}\right) - \frac{\partial^2 G^L}{\partial p_1 \partial p_2} = -1 < 0 \tag{6}$$

Thus, the critical points are saddle points, which can be illustrated in Figure 1. Thus, the minimum value of $G^L(p_1, p_2)$ can be found on the boundary. As there

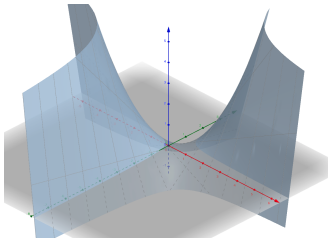

Fig. 1: Saddle surface

exists a point $(p_1^0, p_2^0) \in [l_1, u_1] \times [l_2, u_2]$ such that $G^L(p_1^0, p_2^0) = 0$. Thereby, we need to satisfy constraints on $G^L(p_1, p_2)$ that $G^L(p_1^0, p_2^0) = 0$ and the value of G^L on boundary points: $G^L(l_1, u_1), G^L(l_1, u_2), G^L(l_2, u_1), G^L(l_2, u_2) \geq 0$. Equivalently, this can be formulated as:

$$\begin{cases} c = p_1^0 p_2^0 - a^L p_1^0 - b^L p_2^0 \\ l_1 l_2 - a^L \left(l_1 - p_1^0\right) - b^L \left(l_1 - p_2^0\right) - p_1^0 p_2^0 \geq 0 \\ l_1 u_2 - a^L \left(l_1 - p_1^0\right) - b^L \left(u_2 - p_2^0\right) - p_1^0 p_2^0 \geq 0 \\ u_1 l_2 - a^L \left(u_1 - p_1^0\right) - b^L \left(l_2 - p_2^0\right) - p_1^0 p_2^0 \geq 0 \\ u_1 u_2 - a^L \left(u_1 - p_1^0\right) - b^L \left(u_2 - p_2^0\right) - p_1^0 p_2^0 \geq 0 \end{cases} \tag{7}$$

By substituting the c^L in the equation (26) with equation (30), we can obtain:

$$F^L(p_1, p_2) = \Lambda[(l_1 + u_1 - 2p_1^0)a^L + (l_2 + u_2 - 2p_2)b^L + 2p_1^0 p_2^0],$$

where $\Lambda = \frac{(u_1 - l_1)(u_2 - l_2)}{2}$.

As the minimum value can only be found on the boundary, the minimum value (x_0, y_0) can only be chosen from the border. Thus, the p_1^0 can be l_1 or u_1.

- If $p_1^0 = l_1$:
  By substituting it in equation (8), we get:

$$\begin{cases} a^L \leq \frac{u_1 l_2 - l_1 p_2^0 - b^L \left(l_2 - p_2^0\right)}{u_1 - l_1} \\ a^L \leq \frac{u_1 u_2 - l_1 p_2^0 - b^L \left(u_2 - p_2^0\right)}{u_1 - l_1} \\ l_1 \leq b^L \leq l_1 \Leftrightarrow b^L = l_1 \end{cases} \tag{8}$$

  And also:

$$F^L(p_1, p_2) = \Lambda[(u_1 - l_1)a^L + (l_2 + u_2 - 2p_2^0)b^L + 2l_1 p_2^0]. \tag{9}$$

  Next, replacing $b^L = l_1$ in equation (32), we can obtain:

$$F^L(p_1, p_2) = \Lambda[(u_1 - l_1)a^L + l_1(l_2 + u_2)],$$

and

$$\begin{aligned}(u_1 - l_1)\, a^L &\leq -l_1 p_2^0 + \min\left(u_1 l_2 - b^L\left(l_2 - p_2^0\right), u_1 u_2 - b^L\left(u_2 - p_2^0\right)\right) \\ &= (u_1 - l_1)\min\left(l_2, u_2\right) \\ &= (u_1 - l_1)\, l_2.\end{aligned} \tag{10}$$

As a result, we obtain $a^L \leq l_2$. To maximize $F^L(p_1, p_2)$, because $\Lambda \geq 0$, we adapt $a^L = l_2$ to get $F^L(p_1, p_2) = \Lambda(u_1 l_2 + l_1 u_2)$, which is constant.

- If we choose $p_1^0 = u_1$:
  Similarly, we can compute

$$\begin{cases} a^L \geq \frac{l_1 l_2 - u_1 p_2^0 - b^L\left(l_2 - p_2^0\right)}{l_1 - u_1} \\ a^L \geq \frac{l_1 u_2 - u_1 p_2^0 - b^L\left(u_2 - p_2^0\right)}{l_1 - u_1} \\ u_1 \leq b^L \leq u_1 \Leftrightarrow b^L = u_1 \end{cases} \tag{11}$$

  and

$$F^L(p_1, p_2) = \Lambda\left[\left(l_1 - u_1\right) a^L + \left(l_2 + u_2 - 2p_2^0\right) b^L + 2u_1 p_2^0\right]$$

  By replacing $b^L = u_1$, we obtain

$$\begin{aligned} F^L(p_1, p_2) &= \Lambda\left[\left(l_1 - u_1\right) a^L + u_1\left(l_2 + u_2\right)\right] \\ \left(l_1 - u_1\right) a^L &\leq -u_1 p_2^0 + \min\left(l_1 l_2 - b^L\left(l_2 - p_2^0\right), l_1 u_2 - b^L\left(u_2 - p_2^0\right)\right) \\ &= \left(l_1 - u_1\right)\max\left(l_2, u_2\right) \\ &= \left(l_1 - u_1\right) u_2 \end{aligned}$$

  Then, we can conclude that $a^L \geq u_y$, and $F^L = \Lambda(l_1 u_2 + u_1 l_2)$

As the F^L are the same for above two cases, and it is not determined with p_2^0, we will choose $p_2^0 = l_2$. Thus we can obtain the parameters to compute the lower bounds:

$$\begin{cases} a^L = l_2 \\ b^L = l_1 \\ c^L = -l_1 l_2 \end{cases}$$

C.2 derivation for the upper bound

Similarly, to compute the upper bound, we can formulate the $G^U(p_1, p_2)$ as:

$$G^U(p_1, p_2) = \left(a^U p_1 + b^U p_2 + c^U\right) - p_1 p_2 \tag{12}$$

Thus, the objective function to compute the upper bound is:

$$\min_{a^U, b^U, c^U} F^U(p_1, p_2) = \min_{a^U, b^U, c^U} \int_{p_1 \in [l_1, u_1]} \int_{p_2 \in [l_2, u_2]} a^U p_1 + b^U p_2 + c^U, \tag{13}$$

and then it can be expressed as:

$$F^U(p_1,p_2)=\Lambda[(l_1+u_1-2p_1^0)a^U+(l_2+u_2-2p_2)b^U+2p_1^0p_2^0],$$

where $\Lambda=\frac{(u_1-l_1)(u_2-l_2)}{2}$.

- If we set $p_1^0=l_1$: We obtain the constraints as:

$$\begin{cases} a^U \geq \frac{u_1l_2-l_1p_2^0-b^U\left(l_2-p_2^0\right)}{u_1-l_1} \\ a^U \geq \frac{u_1u_2-l_1p_2^0-b^U\left(u_2-p_2^0\right)}{u_1-l_1} \\ l_1\leq b^U\leq l_1 \Leftrightarrow b^U=l_1 \end{cases} \tag{14}$$

and

$$F^U(p_1,p_2)=\Lambda[(u_1-l_1)a^U+(l_2+u_2-2p_2^0)b^U+2l_1p_2^0]. \tag{15}$$

After substituting $b^U=l_1$,

$$F^U(p_1,p_2)=\Lambda[(u_1-l_1)a^U+l_1(l_2+u_2)],$$

and

$$\begin{aligned}(u_1-l_1)\,a^U &\geq -l_1p_2^0+\max\left(u_1l_2-b^U\left(l_2-p_2^0\right),u_1u_2-b^U\left(u_2-p_2^0\right)\right)\\ &=(u_1-l_1)\max\left(l_2,u_2\right)\\ &=(u_1-l_1)\,u_2.\end{aligned}$$

$$a^U\geq u_2. \tag{16}$$

To minimize $F^U(p_1,p_2)$, we adapt $a^U=u_2$ to get $F^U(p_1,p_2)=\Lambda(l_1l_2+u_1u_2)$.

- If we set $p_1^0=l_1$:

$$\begin{cases} a^U \leq \frac{l_1l_2-u_1p_2^0-b^U\left(l_2-p_2^0\right)}{l_1-u_1} \\ a^U \leq \frac{l_1u_2-u_1p_2^0-b^U\left(u_2-p_2^0\right)}{l_1-u_1} \\ u_1\leq b^U\leq u_1 \Leftrightarrow b^U=u_1 \end{cases} \tag{17}$$

and

$$F^U(p_1,p_2)=\Lambda\left[(l_1-u_1)\,a^U+\left(l_2+u_2-2p_2^0\right)b^U+2u_1p_2^0\right]$$

Next,

$$\begin{aligned}F^U(p_1,p_2)&=\Lambda\left[(l_1-u_1)\,a^U+u_1\left(l_2+u_2\right)\right]\\ (l_1-u_1)\,a^U &\leq -u_1p_2^0+\max\left(l_1l_2-b^U\left(l_2-p_2^0\right),l_1u_2-b^U\left(u_2-p_2^0\right)\right)\\ &=(l_1-u_1)\min\left(l_2,u_2\right)\\ &=(l_1-u_1)\,l_2\end{aligned}$$

Thus, we can conclude that $a^U\leq l_2$, and to minimize F^U, we set$a^U=l_2$ to form $F^U=\Lambda(l_1l_2+u_1u_2)$.

As the two results for F^U are the same, we choose $p_2^0 = l_2$ to obtain the final parameters for the upper bound:

$$\begin{cases} a^U = u_2 \\ b^U = l_1 \\ c^U = -l_1 u_2 \end{cases}$$

References

Shi Z, Zhang H, Chang KW, et al (2020) Robustness Verification for Transformers. In: International Conference on Learning Representations, URL https://openreview.net/forum?id=BJxwPJHFwS