The goal is to prove, for a given network, a certain property will always hold for a range of inputs.

\[ x_0 \in C \]

Input in a given domain,

\[ \hat{x}_n = f(x_0) \]

\(\hat{x}_n\) is the corresponding output.

\[ P(\hat{x}_n) \]

Property \( P \) holds.

Example: Adversarial Example

For each domain, compute lower bounds and upper bounds over the global minimum. Maintain the best upper bound found and split the domains so as to improve the relaxation and derive better bounds. Prune out the domains that can be inferred to not contain the global minimum.

Verification as Optimization

If property is linear: \( P(\hat{x}_n) \triangleq c^T \hat{x}_n > b \)

\[ \min c^T \hat{x}_n - b \]

such that \( l_0 \leq x_0 \leq u_0 \)

\[ \hat{x}_{i+1} = W_{i+1}x_i + b_{i+1} \]

\(\forall i \in \{0, n-1\}\)

\[ x_i = \max(\hat{x}_i, 0) \]

\(\forall i \in \{1, n-1\}\)

Global minimum negative \( \implies \) Counterexample.

Global minimum positive \( \implies \) Property holds.

Can use Maxpooling unit and equivalent network to deal with more complex properties.

OR: \( P(\hat{x}_n) \triangleq \bigvee_i \left[ c^T \hat{x}_n > b \right] \)

\[ \implies \max_i (c^T \hat{x}_n - b) > 0 \]

AND: \( P(\hat{x}_n) \triangleq \bigwedge_i \left[ c^T \hat{x}_n > b \right] \)

\[ \implies - \left( \max_i (-c^T \hat{x}_n + b) \right) > 0 \]

Baseline solution: Mixed Integer Programming

\[ x_i = \max(\hat{x}_i, 0) \]

\(\delta_i \in \{0, 1\}^h\)

\[ x_i \geq 0, \quad x_i \leq u_i \cdot \delta_i \]

\[ x_i \geq \hat{x}_i, \quad x_i \leq \hat{x}_i - (1 - \delta_i) \]

Explaining other methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Branching Function</th>
<th>compute_UB</th>
<th>pick_out</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reluplex [1]</td>
<td>ReLU splitting</td>
<td>Linear constraints + interval analysis bound</td>
<td>Local search by ignoring non-linearities, greedily fixing them, prioritizing most frequently violated ReLUs</td>
<td></td>
</tr>
<tr>
<td>Planet [2]</td>
<td>ReLU splitting</td>
<td>ReLU convex hull built once</td>
<td>SAT solver with clause analysis driven</td>
<td></td>
</tr>
<tr>
<td>ReluBaB</td>
<td>ReLU splitting</td>
<td>ReLU convex hull rebuilt per iteration</td>
<td>Prioritize ReLU of earlier layers</td>
<td></td>
</tr>
<tr>
<td>BaB</td>
<td>Input dimension splitting</td>
<td>ReLU convex hull rebuilt per iteration</td>
<td>Longest edge heuristic</td>
<td></td>
</tr>
<tr>
<td>BaBSB</td>
<td>Input dimension splitting</td>
<td>ReLU convex hull rebuilt per iteration</td>
<td>“Strong branching” simulation using [3] for fast relaxation evaluation</td>
<td></td>
</tr>
</tbody>
</table>

Algorithm 1 Branch and Bound

1. \( \text{BaB}(\text{net}, \text{domain}, i) \)
2. \( \text{global}_{ub} \leftarrow \text{inf} \)
3. \( \text{global}_{lb} \leftarrow -\text{inf} \)
4. \( \text{doms} \leftarrow [(\text{global}_{lb}, \text{global}_{ub})] \)
5. while \( \text{global}_{ub} - \text{global}_{lb} > \epsilon \) do
6. \((\text{dom}, \epsilon) \leftarrow \text{pick}_\text{out}(\text{doms}) \)
7. \( \text{subdom}_{i}, \ldots, \text{subdom}_{a} \leftarrow \text{split}(\text{dom}) \)
8. for \( i = 1 \ldots a \) do
9. \( \text{dom}_{ub} \leftarrow \text{compute_UB}(\text{net}, \text{subdom}_i) \)
10. \( \text{dom}_{lb} \leftarrow \text{compute_UB}(\text{net}, \text{subdom}_i) \)
11. if \( \text{dom}_{ub} < \text{global}_{ub} \) then
12. \( \text{global}_{ub} \leftarrow \text{dom}_{ub} \)
13. else if \( \text{dom}_{lb} < \text{global}_{lb} \) then
14. \( \text{doms} \leftarrow \text{append}(\text{doms}, \text{dom}_{lb}, \text{dom}_{ub}) \)
15. if \( \text{dom}_{lb} < \text{global}_{lb} \) then
16. \( \text{doms} \leftarrow \text{append}(\text{doms}, \text{dom}_{lb}, \text{dom}_{ub}) \)
17. end if
18. end if
19. end for
20. end while
21. return \( \text{global}_{ub} \)
22. end function

Impact of Rebuilding Convex Hull

(a) Approximation on a CollisionDetection net
(b) Approximation on a deep net from ACAS

Benchmarks

CollisionDetection: 500 properties, 6 inputs, 40 ReLU, 4-Maxpool, 19 ReLU, 2 outputs.
ACAS Dataset: 188 properties, 5 inputs, 6 hidden layers of 50 units, 5 outputs.

Runtime Impact of Architecture

(a) Network Depth
(b) Width of the Network
(c) Number of Inputs
(d) Property Margin

PCAMNIS: Networks taking the first inp components of a PCA on MNIST, with depth hidden layers of width hidden layers. The property is on whether there exist an input leading to a gap superior to a certain threshold between the outputs. The threshold is chosen to change the margin by how much the property is SAT / UNSAT.