Sherlock - Output range analysis for DNN

Monday, 26 October 2020 11:52

- The goal: developing a technique to verify NNs to check wether certain properties are satisfied.
- The verification problem: computing a guaranteed range of the output of a DNN given a set of inputs represented as a convex polyhydron.
  → find the maximum and minimum value taken by the outputs of the NN over the given input set using a combination of ① local search and ② linear programming.
- The range estimation problem:
  given a neural network N and a polyhedron $\phi(x)$ representing a set of inputs to the network, we wish to estimate a range $\text{range}(\hat{l}_i, \phi)$ for each of the network's output $\hat{l}_i$ that subsumes all possible outputs and is tight within a given tolerance δ.
- Sherlock works with ① feed-forward NN with ② ReLU as activation function.
- Finding out of bounds outputs through verification of CPS allows design-time detection of potential failures instead of relying on runtime monitoring.
- Sherlock: is an implementation of an algorithm for propagating convex polyhedral inputs through a feedforward DNN with ReLU activation function to establish ranges for the output of the network.
- The function F computed by a NN N is ① continuous, ② piecewise affine and ③ differentiable almost everywhere in $\mathbb{R}^N$.
  → the gradient (if exists) ∇F_i: $(\partial_{x_1} F_i, \dots, \partial_{x_n} F_i)$
- Mixed Integer Linear Program (MILP):
  - involves a set of real-valued variables x and integer-valued variables w and has the following form:

$$\begin{aligned} \max \quad & a^T x + b^T w \\ \text{s.t.} \quad & Ax + Bw \le c \\ & x \in \mathbb{R}^N,\ w \in \mathbb{R}^m \end{aligned}$$

  - Special cases: 1. if there are no integer variables w
    → the problem is called a Linear program (LP)
    2. if there is no explicit objective function
    → MILP feasibility problem.
  - MILPs are NP-hard problems (exponential time complexity in worst case) but LPs can be solved efficiently using interior point methods.
- Problem definition:
  Let N be a neural network with n inputs x a single output y and weights $\langle (W_i, b_i) \rangle_{i=0..k}$
  Let F_N be the function defind by such network;
  Range estimation problem is defined as follows:
  Inputs: ① Neural network N, ② input constraints P: $Ax \le b$ and ③ a tolerance $\delta > 0$.
  Output: An interval $[l, u]$ such that:
  1. contains the range of F_N over $x \in P$ ($\forall x \in P$) $F_N(x) \in [l, u]$
  2. is tight: ($\max_{x \in P} F_N(x) \ge u - \delta$) and ($\min_{x \in P} F_N(x) \le l + \delta$)
  - We assume the input polyhedron P is compact
    i.e. is ① closed and ② has a bounded volume.
- Overall approach:
  we focus on estimating the upper bound u (the case for l is analogous).
  Problem: a single MILP can be used to directly compute u but it is quite expensive in practice.
  Solution: combine ① a sieries of MILP feasibility problems with ② local search steps.

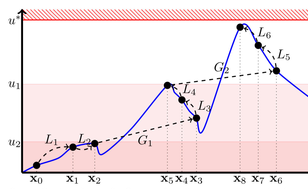

Figure 2: A schematic figure showing our approach showing alternating series of local search $L_1, \dots, L_6$ and "global search" G_1, G_2 iterations. The points x_2, x_5, x_8 represent local minima wherein our approach transitions from local search iterations to global search iterations.

- The Algorithm; finds incrementally a series of $u_1 < u_2 < \dots < u^*$

Algorithm 1 Estimate maximum value u for a neural network $\mathcal{N}$ over a range $x \in P$ with tolerance $\delta > 0$.

```
1:  procedure FindUpperBound(N, P, δ)
2:      x ← Sample(P)                          ← x0 ∈ P
3:      terminate ← false                      ← LP interior point solver
4:      while not terminate do                 ← u = F_N(x) then gradient ascent
5:          (x, u) ← LocalSearch(N, x, P)
6:          u ← u + δ
7:          (x', u', feasible) ← GlobalSearch(N, u, P)   → MILP feasibility problem
8:          if feasible then                   ← new witness point x_{i+1} s.t. F_N(x_{i+1}) ≥ u_i + δ
9:              (x, u) ← (x', u')              to check if there is any point x' ∈ P s.t. F_N(x') ≥ u' + δ
10:         else                               (a better point)
11:             terminate ← true
12:         end if
13:     end while
14:     return (x, u)
15: end procedure
```

  - u increases by at least δ each iteration
    → the upper bound for the number of steps $\left\lceil \frac{(u^* - u_0)}{\delta} \right\rceil$
- Local search:
  - Technique: x_{i+1} is obtained from x_i as follows:
    1. compute the gradient J: $\nabla F_N(x_i)$
    2. compute a locally active region $\hat{L}(x_i)$
    3. Solve a linear program (LP) to compute x_{i+1}
  - Gradient calculation:
    - The gradient doesn't exist for a set of points of measure 0.
    - computed by the chain rule J: $J_0 \times J_1 \times \dots \times J_k$
      J_i is the Jacobian matrix of partial derivatives of the $i+1$ layer z_{i+1} w.r.t. those of the i^{th} layer z_i
      since $z_{i+1} = \sigma(W_i z_i + b_i) \Rightarrow J_i = \begin{cases} W_{i,(j,i)} & ; z_{i+1,j} \ge 0 \\ 0 & \end{cases}$
  - Locally active region: $\hat{L}(x)$
    - describes the set of all inputs x' s.t. x' activates exactly the same neurons as x.
      → $\hat{L}(x)$ is described by a polyhedron with possibly strict inequality constraints. If $x' \in \hat{L}(x)$ then $\nabla F_N(x) = \nabla F_N(x')$.
    - Let $\bar{L}(x)$ denote the closure of the local active set by converting the strict $>$ constraints to their non-strict $\ge$ versions.
      → The local minimum is simply obtained by solving the following LP:

$$\max \ w^T y \quad \text{s.t.} \quad y \in \bar{L}(x) \cap P$$

      the solution of the LP yields a step of the local search.
- Global search:
  - the goal is to search for a point $x \in P$ s.t. $F_N(x) \ge u$ for a given estimate u of the current upper bound.
  - we have x inputs, y output, $z_1, \dots, z_{k-1}$ outputs of the hidden layers $t_1, \dots, t_{k-1}$ binary variables to model the piecewise behavior of ReLU
    M is a very large constant (∞)
    as $z_{i+1} = \sigma(W_i z_i + b_i)$ we use t_i to encode the behavior of P: $Ax \le b$

$$\begin{aligned} z_{i+1} &\ge W_i z_i + b_i \\ z_{i+1} &\le W_i z_i + b_i + M t_{i+1} \\ z_{i+1} &\ge 0 \\ z_{i+1} &\le M(1 - t_{i+1}) \end{aligned}$$

    → the resulted problem is an MILP feasibility problem without any constraints.
  - The MILP encoding is feasible if and only if there is an input $x \in P$ s.t. $y = F_N(x) \ge u$.
  - MILP solver Gurobi optimization ['16]