Reliable and Interpretable Artificial Intelligence Project

Hsuan-I Ho (hohs@student.ethz.ch), Chi-Ching Hsu (hsuch@student.ethz.ch)

In this project, we want to (1) verify the robustness of neural networks as sound as possible via Gurobi linear solver while (2) complete a single verification task within 7-minute timeout. Therefore, we first analyzed the running time exploiting linear solver for all layers during the verification and figured out that except **mnist_relu_4_1024**, all other provided networks can be verified within 7 minutes (with test image: img66, epsilon=0.01, running on 2.5GHz i5-7200 CPU). In the next step, we then layer by layer replaced the linear solver by interval analysis as provided in the skeleton. Fig. 1 shows the results of running time and output bounds in the considered verification settings. We can see that any replacement of the interval analysis cause failed verification due to the lack of precision and significant changes in output bounds. In addition, we also figured out that linear solver should never be used in the low-level layers (layers that close to the input layer) since it not only loses precision but also increases overall running time. We will follow this rule of thumb when the neural network becomes larger and can no longer be verified within the time limit.

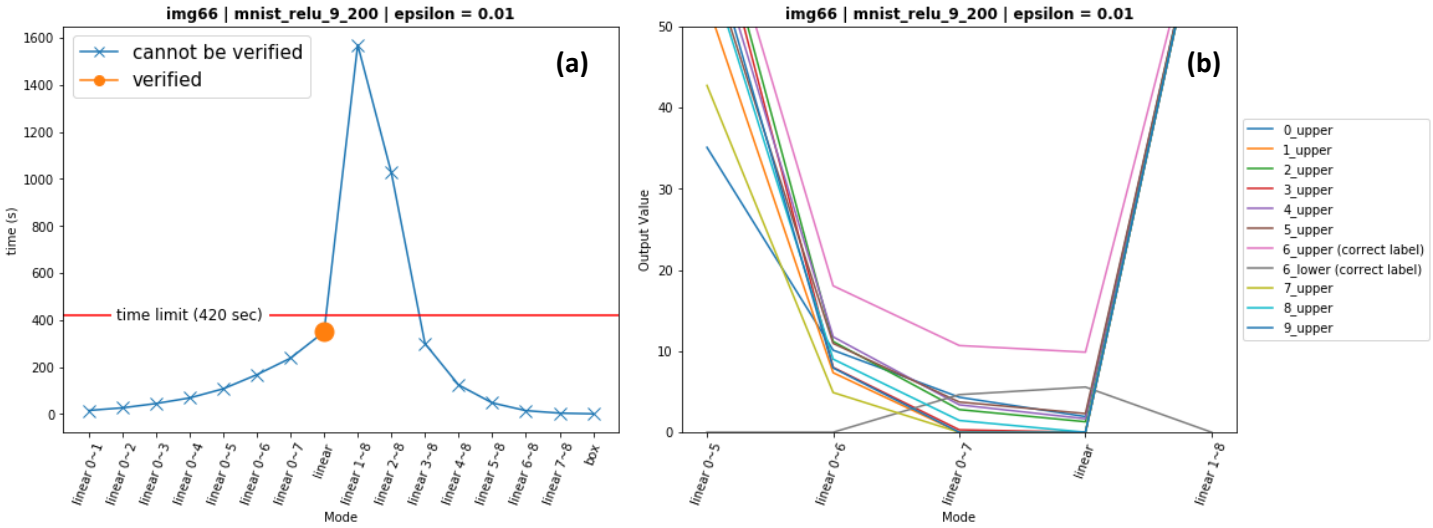

Fig. 1 (a) Running time and (b) output bounds of the considered verification settings. The indices in x-axis after "linear" indicate in which network layer we apply the linear solver. "**Linear**" represents applying linear solver to all network layers (can be served as the ground truth), while "**Box**" represents only interval analysis (the most imprecise result).

In addition, we further analyzed the correlation between running time and epsilon. In general, the larger the epsilon is, the more constraints and variables are added to the solver and thus more computation time is required. However, the running time also depends on the input images and not always satisfies this correlation. We thus take both the number of hidden units (variables to be solved) in the network and the input epsilon into account when deciding our verification setting.

As for optimizing the computation of the linear solver, we have designed the following improvement. First, we jointly update the intervals and variables constraints based on the results of linear solver. This strategy helps us reduce the searching space when applying the solver. Secondly, for the networks that is too large to add constraints and variables, we designed a dynamic programming based linear solver approach to reduce the computation. In particular, we store ReLU output as a linear combination of the input variables in a data structure, and only use the solver when solving the objective functions. However, this can also cause precision loss since the output bounds are not as tight as the bound obtained by using only linear solver.

In summary, we decide the verification setting based on the following rules of thumb: 1) for the network consists less hidden units, we apply linear solver to all layers; 2) for the network consists more hidden units (about 1500 units with a large epsilon based on our experiment), we implement dynamic programming based linear solver method; 3) when the computation bottleneck of dynamic programming reaches the time limit, we start to apply interval analysis in the high-level layers (layers that close to outputs). In particular, we only replace at most one layer by interval analysis due to the precision loss. Finally, since the machine used for grading is different from our test environment, we might need to slightly adjust the above parameters according to the evaluation results from TA.