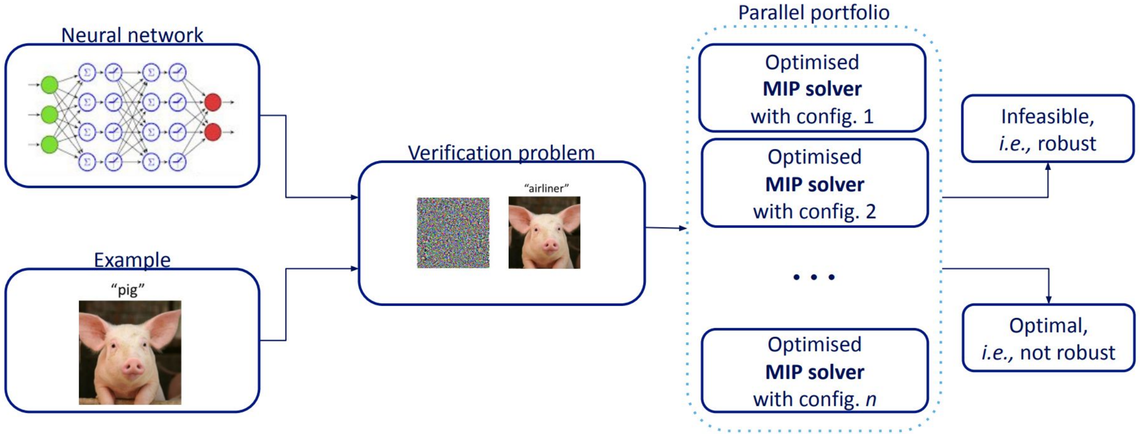
Performance regression and algorithm selection for MIP-based Neural Network Verification

Jasmin Kareem
MSc Data Science: Computer Science Student at Leiden University

Introduction

- Neural Network Verification using a parallel portfolio of solver configurations
- Reduces number of timeouts and improves verification speed
- Further improvements to performance are possible.
- Two main goals:
  - Predicting performance (runtime) of the configurations (with the use of problem-specific instance features)
  - Implementing per-instance algorithm selection techniques to this problem



Source: König, M., Hoos, H.H. & Rijn, J.N.v. Speeding up neural network robustness verification via algorithm configuration and an optimised mixed integer linear programming solver portfolio

Predicting performance of configurations

- Configurations have different solve times, some even timeout

- Can we predict performance?

Config	Solve time	Status
1	2456.4	Infeasible/Unbounded
2	2176.09	Infeasible
3	3692.77	Infeasible
default	9600	User limit

Example of a single sample of the output of the MIPverify solver configurations (sample 17).

Finding features for MIP based problems

- Work already exists for MIP instance features

- But there are other ways to find features for this problem

Problem Type (trivial):

1. **Problem type**: LP, MILP, FIXEDMILP, QP, MIQP, FIXEDMIQP, MIQP, QCP, or MIQCP, as attributed by `CPLEX`

Problem Size Features (trivial):

2–3. **Number of variables and constraints**: denoted n and m, respectively
4. **Number of non-zero entries in the linear constraint matrix, $\boldsymbol{A}$**
5–6. **Quadratic variables and constraints**: number of variables with quadratic constraints and number of quadratic constraints
7. **Number of non-zero entries in the quadratic constraint matrix, $\boldsymbol{Q}$**
8–12. **Number of variables of type**: Boolean, integer, continuous, semi-continuous, semi-integer
13–17. **Fraction of variables of type** (summing to 1): Boolean, integer, continuous, semi-continuous, semi-integer
18–19. **Number and fraction of non-continuous variables** (counting Boolean, integer, semi-continuous, and semi-integer variables)
20–21. **Number and fraction of unbounded non-continuous variables**: fraction of non-continuous variables that has infinite lower or upper bound
22–25. **Support size**: mean, median, vc, q90/10 for vector composed of the following values for bounded variables: domain size for binary/integer, 2 for semi-continuous, 1+domain size for semi-integer variables.

Variable-Constraint Graph Features (cheap): each feature is replicated three times, for $X \in \{C, NC, V\}$

26–37. **Variable node degree statistics**: characteristics of vector $(\sum_{c_j \in C} \mathbb{I}(A_{i,j} \neq 0))_{x_i \in X}$: mean, median, vc, q90/10
38–49. **Constraint node degree statistics**: characteristics of vector $(\sum_{x_i \in X} \mathbb{I}(A_{i,j} \neq 0))_{c_j \in C}$: mean, median, vc, q90/10

Linear Constraint Matrix Features (cheap): each feature is replicated three times, for $X \in \{C, NC, V\}$

50–55. **Variable coefficient statistics**: characteristics of vector $(\sum_{c_j \in C} A_{i,j})_{x_i \in X}$: mean, vc
56–61. **Constraint coefficient statistics**: characteristics of vector $(\sum_{x_i \in X} A_{i,j})_{c_j \in C}$: mean, vc
62–67. **Distribution of normalized constraint matrix entries, $\boldsymbol{A}_{i,j}/b_i$**: mean and vc (only of elements where $b_i \neq 0$)
68–73. **Variation coefficient of normalized absolute non-zero entries per row** (the normalization is by dividing by sum of the row's absolute values): mean, vc

Objective Function Features (cheap): each feature is replicated three times, for $X \in \{C, NC, V\}$

74–79. **Absolute objective function coefficients** $\{|c_i|\}_{i=1}^n$: mean and stddev
80-85. **Normalized absolute objective function coefficients** $\{|c_i|/n_i\}_{i=1}^n$, where n_i denotes the number of non-zero entries in column i of $\boldsymbol{A}$: mean and stddev
86-91. **squareroot-normalized absolute objective function coefficients** $\{|c_i|/\sqrt{n_i}\}_{i=1}^n$: mean and stddev

LP-Based Features (expensive):

92–94. **Integer slack vector:** mean, max, L_2 norm
95. **Objective function value of LP solution**

Right-hand Side Features (trivial):

96–97. **Right-hand side for $\leqslant$ constraints**: mean and stddev
98–99. **Right-hand side for $=$ constraints**: mean and stddev
100–101. **Right-hand side for $\geqslant$ constraints**: mean and stddev

Presolving Features* (moderate):

102-103. **CPU times**: presolving and relaxation CPU time
104-107. **Presolving result features:** # of constraints, variables, non-zero entries in the constraint matrix, and clique table inequalities after presolving.

Probing Cut Usage Features* (moderate):

108-112. **Number of specific cuts**: clique cuts, Gomory fractional cuts, mixed integer rounding cuts, implied bound cuts, flow cuts

Probing Result features* (moderate):

113-116. **Performance progress**: MIP gap achieved, # new incumbent found by primal heuristics, # of feasible solutions found, # of solutions or incumbents found

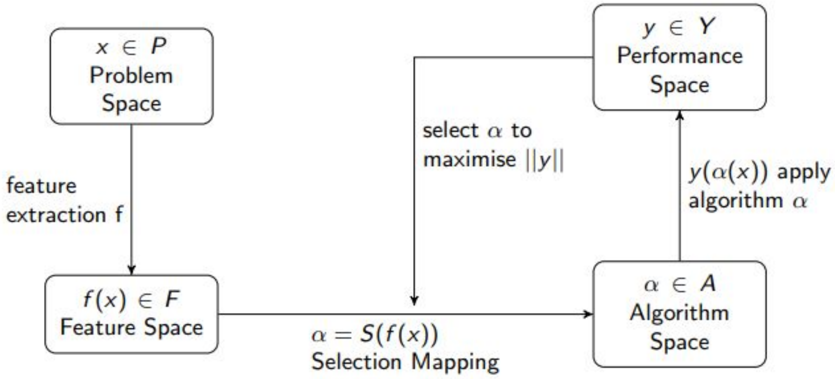
Timing Features*

117–121. **CPU time required for feature computation:** one feature for each of 5 groups of features (see text for details)

Fig. 2. MIP instance features; for the variable-constraint graph, linear constraint matrix, and objective function features, each feature is computed with respect to three subsets of variables: continuous, C, non-continuous, NC, and all, V. Features introduced for the first time are marked with *.

Source: Frank Hutter, Lin Xu, Holger H. Hoos, Kevin Leyton-Brown, Algorithm runtime prediction: Methods & evaluation, Artificial Intelligence

Per-instance algorithm selection



Source: J.R. Rice, The Algorithm Selection Problem, 1976.

Going forward…

1. Make an overview of the current literature on:
   - MIP solvers
   - Problem specific instance features
   - Algorithm selection methods
   - Etc..
2. Replicating results from relevant papers to the problem
3. My own implementation
   - Building model to predict runtime using found features
   - Algorithm selection