PROCEEDINGS OF THE 32$^{\text{nd}}$ European Symposium on Computer Aided Process Engineering
(ESCAPE 32), June 12-15, 2022, Toulouse, France
L. Montastruc, S. Negny (Editors)
© 2022 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/B978-0-323-95879-0.50098-9

Verification of Neural Network Surrogates

Joshua Haddad[a], Michael Bynum[a], Michael Eydenberg[a], Logan Blakely[a],
Zachary Kilwein[b], Fani Boukouvala[b], Carl D. Laird[c] and Jordan Jalving[a]

[a]*Sandia National Laboratories, Albuquerque, NM 87123, USA*
[b]*Department of Chemical and Biomolecular Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA*
[c]*Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA*
jhjalvi@sandia.gov

Abstract

Neural networks (NN)s have been increasingly proposed as surrogates for approximation of systems with computationally expensive physics for rapid online evaluation or exploration. As these surrogate models are integrated into larger optimization problems used for decision making, there is a need to verify their behavior to ensure adequate performance over the desired parameter space. We extend the ideas of optimization-based neural network verification to provide guarantees of surrogate performance over the feasible optimization space. In doing so, we present formulations to represent neural networks within decision-making problems, and we develop verification approaches that use model constraints to provide increasingly tight error estimates. We demonstrate the capabilities on a simple steady-state reactor design problem.

Keywords: Neural Networks, Verification, Optimization, Surrogate Modeling

1. Introduction

Neural networks (NNs) have seen wide success across engineering disciplines. Their excellent approximation qualities (Hornik et al. (1989)) can reduce challenging problems into tractable computational models and their training procedure can incorporate diverse data sets and expert domain knowledge (Beck et al. (2016)). Successful engineering applications have harnessed NNs in planning, design, and control (Pistikopoulos et al. (2021)) with notable process-systems applications that span: forecasting renewable energy generation (Lee et al. (2016)), predicting distillation dynamics (Sánchez-Ramírez et al. (2020)), and estimating reactor performance (Salah et al. (2016)).

Typical NN applications are concerned with making forward predictions (i.e., NNs are evaluated in a forward-mode), but they are also advantageous in the context of optimization where they can take on the form of *algebraic surrogates*. Such surrogate models can be used to reduce the complexity of underlying equations by replacing them with more tractable formulations. Neural network surrogates have seen successful application in optimizing chemical process operation (Fernandes (2006)), performing process synthesis with super-structures (Henao and Maravelias (2010)), solving stochastic optimization problems to operate distillation columns (Gutiérrez-Antonio (2016)), and representing contingency constraints in security-constrained optimal power flow (Kilwein et al. (2021)).

The *verification* of neural networks is often concerned with finding adversarial inputs (Goodfellow et al. (2015)) using techniques such as mixed-integer-linear-programming (MILP) (Tjeng et al. (2017)), Satisfiability Modulo Theories (Scheibler et al. (2015)), and Lagrangian duality

(Dvijotham et al. (2018)). This manuscript extends verification concepts and addresses the need to verify NNs that are deployed as optimization-based surrogates. In contrast to methods that verify over a specified input space, we propose formulations that verify over constraints involving both the inputs *and* outputs of the neural network. In doing so, we develop optimization problems that verify worst-case NN prediction error subject to known model physics and operational constraints, and we show that incorporating known constraints leads to tighter error estimates than simply verifying over the input space.

2. Optimization with Neural Network Surrogates

We consider the solution of an optimization problem given by, (1),

$$\begin{aligned} f^*(p) = \min_{x,y,z} \quad & f(x,y,z,p) && \text{(1a)}\\ \text{s.t.} \quad & y = h(x,p) && \text{(1b)}\\ & c(x,y,z,p) = 0 && \text{(1c)}\\ & g(x,y,z,p) \leq 0 && \text{(1d)}\\ & x^L \leq x \leq x^U && \text{(1e)} \end{aligned}$$

where p represents system parameters which are known inputs for a particular optimization instance, and x, y, and z represent optimization variables. We desire efficient solutions of this problem for different values of the parameters p (e.g., in an online context, or for multi-scenario analysis). For improved solution, we consider instead an approximate formulation where we replace a portion of the model with a neural network surrogate. We assume that the NN surrogate brings some benefit to the optimization problem by facilitating a more tractable computation. This could entail the simplification of (1b) to support more rapid or reliable optimization in an online context, or involve creating a piecewise linear approximation (e.g., using ReLU activation functions) of h to facilitate global optimization approaches. We show the new optimization formulation below, where we approximate h in (1b) with the neural network N in (2b). Here, $\hat{y}$ represents the neural network output, and is an approximation of the original variables y.

$$\begin{aligned} f_S^*(p) := \min_{x,\hat{y},z} \quad & f(x,\hat{y},z,p) && \text{(2a)}\\ \text{s.t.} \quad & \hat{y} = N(x,p) && \text{(2b)}\\ & c(x,\hat{y},z,p) = 0 && \text{(2c)}\\ & g(x,\hat{y},z,p) \leq 0 && \text{(2d)}\\ & x^L \leq x \leq x^U && \text{(2e)} \end{aligned}$$

This hybrid modeling approach is common in engineering applications. However, before "deploying" problem (2) in a decision-making application, we wish to verify the accuracy of the neural network approximation. In the next section, we describe a verification formulation that explicitly considers the feasible region of the constraints above and the range on the input parameters p.

3. Formulation of Neural Network Verification Problem

Typical verification approaches seek to determine the maximum error between y and $\hat{y}$ over a predefined input space in x and p. However, these approaches can produce errors that are larger than necessary since they allow points in x and p that may not feasible with respect to the constraints in formulations (1) and (2). Here, we are specifically interested in the accuracy of the neural network over the feasible space of the optimization problem, and we formulate the verification problem (3) shown below:

$$\max_{x,p,y,\hat{y},z} \quad ||y-\hat{y}||_\infty \tag{3a}$$

$$\text{s.t.} \quad y = h(x,p) \qquad \text{True Model} \tag{3b}$$

$$\hat{y} = N(x,p) \qquad \text{NN Surrogate} \tag{3c}$$

$$x^L \le x \le x^U \qquad \text{Input Bounds} \tag{3d}$$

$$p^L \le p \le p^U \qquad \text{Parameter Bounds} \tag{3e}$$

$$\begin{bmatrix} c(x,y,z,p) = 0 \\ g(x,y,z,p) \le 0 \end{bmatrix} \underline{\vee} \begin{bmatrix} c(x,\hat{y},z,p) = 0 \\ g(x,\hat{y},z,p) \le 0 \end{bmatrix} \qquad \text{Feasibility Constraints} \tag{3f}$$

Here, the objective function (3a) maximizes the infinity norm over the NN prediction errors. Maximizing other error measures such as mean-squared-error, mean-absolute-error, or individual prediction errors is also possible. The constraints (3b) and (3c) are the same as (1b) and (2b), respectively, which relate the NN inputs to the true model variables y and the neural network outputs $\hat{y}$. As indicated above, a natural choice of verification constraints are limits on the *input space* of the neural network (i.e., bounds over x and p). While a typical verification formulation would include only (3a)-(3e), we seek to limit the verification to points in x and p that are feasible with respect to the optimization problems (1) and (2).

Our target optimization problem (2) includes feasibility constraints over $\hat{y}$. However, including only constraints (2c-2d) in the verification problem (rather than the disjunction in (3f)) is insufficient because of errors between y and $\hat{y}$. Constraints (2c-2d) alone may not sufficiently represent the feasible region in problem (1). Indeed, if the NN accuracy is low, the feasible region for x and p could be significantly underestimated or even empty. Therefore, we search for the maximum error over both the constraints (1c-1d) and (2c-2d) as represented by the disjunction in (3f). With this, problem (3) finds the maximum deviation between y and $\hat{y}$ subject to the constraints of the optimization problem applied conservatively to either y or $\hat{y}$.

For the case studies in this paper, we consider three formulations to analyze the verification approach. Formulation (V1) represents the typical verification formulation used in the literature which considers only explicit constraints on the input space for x and p. In practice, while we could solve the disjunctive problem (3) directly, it is convenient to solve with each disjunct separately where the solution of (3) is given by the maximum of the solutions from (V2) and (V3) below.

$$\begin{aligned} \max_{x,p,y,\hat{y},z} \quad & ||y-\hat{y}||_\infty \\ \text{s.t.} \quad & (3b)-(3e) \end{aligned} \tag{V1}$$

$$\begin{aligned} \max_{x,p,y,\hat{y},z} \quad & ||y-\hat{y}||_\infty \\ \text{s.t.} \quad & (3b)-(3e) \\ & c(x,y,z,p) = 0 \\ & g(x,y,z,p) \le 0 \end{aligned} \tag{V2}$$

$$\begin{aligned} \max_{x,p,y,\hat{y},z} \quad & ||y-\hat{y}||_\infty \\ \text{s.t.} \quad & (3b)-(3e) \\ & c(x,\hat{y},z,p) = 0 \\ & g(x,\hat{y},z,p) \le 0 \end{aligned} \tag{V3}$$

4. Illustrative Example: Reactor Optimization with Neural Network Surrogates

We provide an illustrative reactor optimization example where we use NNs (with ReLU activations) to replace nonlinear physics with piecewise-linear approximations. We demonstrate global solution of the verification problem and compare with the input-only formulation given by (V1).

4.1. Reactor Optimization Problem

The problem of interest is a steady-state continuous-stirred-tank-reactor (CSTR) that converts feed components A and B to produce D as depicted by Figure 1 where C is an intermediate and E is a

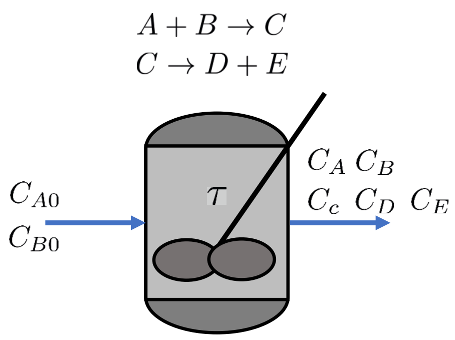


Figure 1: Simple Steady-State CSTR

side product. The overall mass balance is given by (4) (or by (5) in terms of the neural network outputs, $\hat{y}$).

$$C_i = C_{i,0} - \tau_1^r \quad i \in \{A,B\} \tag{4a}$$
$$C_C = C_{C,0} + \tau_1^r - \tau_2^r \tag{4b}$$
$$C_i = C_{i,0} + \tau_2^r \quad i \in \{D,E\} \tag{4c}$$

$$C_i = C_{i,0} - \hat{\tau}_1^r \quad i \in \{A,B\} \tag{5a}$$
$$C_C = C_{C,0} + \hat{\tau}_1^r - \hat{\tau}_2^r \tag{5b}$$
$$C_i = C_{i,0} + \hat{\tau}_2^r \quad i \in \{D,E\} \tag{5c}$$

We define variables τ_1^r and τ_2^r in (6) to represent the product of τ (the space-time) and the corresponding reaction rate (r_1 or r_2). The reaction rate constants K_i are selected as the parameters p. We also include operating constraints given by (7) which correspond to the minimum conversion and yield the reactor must achieve with η_1=0.9 and η_2=0.2.

$$\tau_1^r = \tau r_1 = \tau K_1 C_A C_B \tag{6a}$$
$$\tau_2^r = \tau r_2 = \tau K_2 C_c \tag{6b}$$

$$C_D \geq \eta_1(C_{A,0} - C_A) \tag{7a}$$
$$C_D \geq \eta_2(C_A + C_B + C_c) \tag{7b}$$

The true design problem as a function of p is then given by (8) where we seek to minimize the space-time τ (i.e., maximize throughput) subject to physical balances and operating requirements. Consequently, we train ReLU-based NNs to replace the associated nonlinear terms and formulate the approximate design problem (9) as an MILP (Grimstad and Andersson (2019)).

$$\min_{\tau} \quad \tau \tag{8a}$$
$$\text{s.t.} \quad \tau_1^r = \tau K_1 C_A C_B \tag{8b}$$
$$\tau_2^r = \tau K_2 C_c \tag{8c}$$
$$(4), (7)$$

$$\min_{\tau} \quad \tau \tag{9a}$$
$$\text{s.t.} \quad \hat{\tau}_1^r = K_1 N_1(C_A, C_B, \tau) \tag{9b}$$
$$\hat{\tau}_2^r = K_2 N_2(C_C, \tau) \tag{9c}$$
$$(5), (7)$$

4.2. Verification Problem

The verification problem corresponding to (3) is given by (11) with the NN input and model parameter bounds defined by (10) below.

$$\begin{aligned} &C_A^L \leq C_A \leq C_A^U, \quad C_B^L \leq C_B \leq C_B^U, \quad C_C^L \leq C_C \leq C_C^U, \\ &\tau^L \leq \tau \leq \tau^U, \quad K_1^L \leq K_1 \leq K_1^U, \quad K_2^L \leq K_2 \leq K_2^U \end{aligned} \tag{10}$$

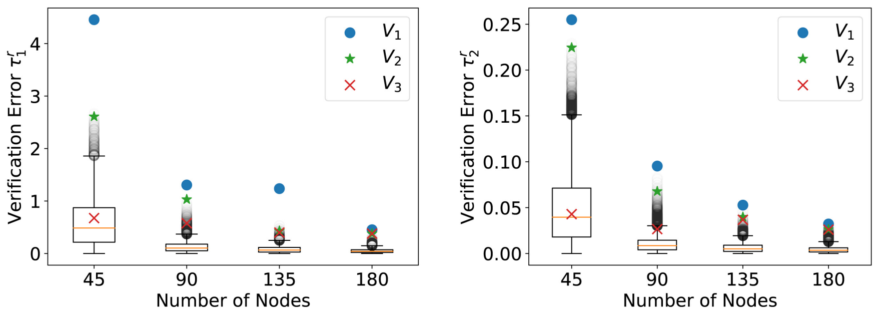
The valid range for the parameters K is given by K_1=0.31051 $\pm$ 10% $\frac{\ell}{mol-s}$ and K_2=0.026650 $\pm$ 10% $\frac{1}{s}$. For this example we seek to maximize the squared prediction error of both τ_1^r and τ_2^r (i.e., we solve verification problems for each output) but it is also possible to formulate mean-absolute error using integer variables.

$$
\begin{aligned}
\max \quad & (\hat{\tau}_j^r - \tau_j^r)^2 \quad j \in \{1,2\} && && \text{(11a)}\\
\text{s.t.} \quad & (8b),\ (8c) && \text{Original Model} && \text{(11b)}\\
& (9b), (9c) && \text{NN} && \text{(11c)}\\
& (10) && \text{Input Bounds and Parameter Bounds} && \text{(11d)}\\
& \big[(4),\ (7)\big] \veebar \big[(5),\ (7)\big] && \text{Model Constraints OR NN Constraints} && \text{(11e)}
\end{aligned}
$$

We apply each of the presented verification formulations to our reactor example. In particular, we formulate (V1) with equations (11a-11d). We formulate (V2) and (V3) using these equations along with each of the individual disjuncts in (11e).

4.3. Results

We use `TensorFlow 2.3` to train the multi-layer neural network surrogates with increasing numbers of nodes using ReLU activation functions. We use `Gurobi 9.1` to solve the non-convex true design problem (8) and the verification problems. The results are presented in Figure 2 and are summarized as follows: (i) (V1) provides the expected global worst-case error over the input space, but we can obtain tighter error bounds using (V2) and (V3) which satisfy the known constraints, (ii) (V3) is tighter than (V2) with smaller networks where the accuracy is poor and the feasible region is poorly approximated, and (iii) worst-case error improves for larger neural networks, but there are likely trade-offs with performance vs accuracy. We also compare the solution



(a) (b)

Figure 2: Results for test error (box plots) and verification (colored markers for V1, V2, and V3). Observed errors for τ_1^r (left) and τ_2^r (right) for increasingly larger neural networks.

of (9) for each NN to the true problem (8) using `Gurobi 9.1`. Table 1 shows each NN prediction and objective value τ which are consistent with the verification findings. Tighter worst-case error estimates correspond with closer approximations of the true problem, but even NNs with considerable error according to (V1) perform adequately as algebraic surrogates when considering the feasible region and their usage within the optimization problem.

5. Conclusions

This manuscript explores verification approaches for neural network (NN) surrogates used within optimization problems. We demonstrated how verification over known constraints produces tighter worst-case NN violations. We presented an illustrative reactor design example to elucidate verification concepts for NNs used as surrogates in an optimization setting.

Table 1: Comparison of results for reactor optimization problem solution for increasingly larger neural networks. Percentages correspond to percent difference from the true global solution.

		Error (Relative Error %)							
Variable	True	45 Nodes		90 Nodes		135 Nodes		180 Nodes	
τ_1^r	0.527	-3.9E-3	(-0.75%)	1.5E-3	(0.28%)	-8.8E-4	(-0.17%)	5.6 E-4	(0.11%)
τ_2^r	0.474	-3.5E-3	(-0.75%)	1.3E-3	(0.28%)	-7.9E-4	(-0.17%)	5.0 E-4	(0.11%)
τ	337.7	-14.5	(-4.29%)	-10.7	(-3.17%)	3.3	(0.99%)	0.17	(0.05%)

6. Acknowledgements

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government. This work was funded in part by the Institute for the Design of Advanced Energy Systems (IDAES) with funding from the Office of Fossil Energy, Cross-Cutting Research, U.S. Department of Energy. This work was also funded by Sandia National Laboratories Laboratory Directed Research and Development (LDRD) program.

References

D. A. C. Beck, J. M. Carothers, V. R. Subramanian, J. Pfaendtner, 2016. Data Science : Accelerating Innovation and Discovery in Chemical Engineering 62 (5).

K. Dvijotham, R. Stanforth, S. Gowal, T. A. Mann, P. Kohli, 2018. A dual approach to scalable verification of deep networks. Proceedings of the Thirty-Fourth Conference Annual Conference on Uncertainty in Artificial Intelligence.

F. A. N. Fernandes, 2006. Optimization of fischer-tropsch synthesis using neural networks. Chemical Engineering & Technology 29 (4), 449–453.

I. J. Goodfellow, J. Shlens, C. Szegedy, 2015. Explaining and harnessing adversarial examples. In: International Conference on Learning Representations.

B. Grimstad, H. Andersson, 2019. ReLU networks as surrogate models in mixed-integer linear programs. Computers and Chemical Engineering 131, 106580.

C. Gutiérrez-Antonio, 01 2016. Multiobjective stochastic optimization of dividing-wall distillation columns using a surrogate model based on neural networks. Chemical and Biochemical Engineering Quarterly 29, 491–504.

C. A. Henao, C. T. Maravelias, 2010. Surrogate-based process synthesis. Computer-aided chemical engineering 28.

K. Hornik, M. Stinchcombe, H. White, 1989. Multilayer feedforward networks are universal approximators. Neural Networks 2 (5), 359–366.

Z. Kilwein, F. Boukouvala, C. Laird, A. Castillo, L. Blakely, M. Eydenberg, J. Jalving, L. Batsch-Smith, 01 2021. AC-Optimal Power Flow Solutions with Security Constraints from Deep Neural Network Models. Vol. 50.

S. Lee, J.-H. Ryu, B.-M. Hodge, I.-B. Lee, 2016. Development of a neural network-based renewable energy forecasting framework for process industries. In: Z. Kravanja, M. Bogataj (Eds.), 26th European Symposium on Computer Aided Process Engineering. Vol. 38 of Computer Aided Chemical Engineering. Elsevier, pp. 1527–1532.

E. N. Pistikopoulos, A. Barbosa-povoa, J. H. Lee, R. Misener, A. Mitsos, G. V. Reklaitis, V. Venkatasubramanian, F. You, R. Gani, 2021. Process systems engineering – The generation next ? Computers and Chemical Engineering 147.

A. Salah, L. Hanel, M. Beirow, G. Scheffknecht, 2016. Modelling ser biomass gasification using dynamic neural networks. In: Z. Kravanja, M. Bogataj (Eds.), 26th European Symposium on Computer Aided Process Engineering. Vol. 38 of Computer Aided Chemical Engineering. Elsevier, pp. 19–24.

K. Scheibler, L. Winterer, R. Wimmer, B. Becker, 2015. Towards verification of artificial neural networks. In: MBMV.

E. Sánchez-Ramírez, J. G. Segovia-Hernández, E. A. Hernández-Vargas, 2020. Artificial neural network to capture the dynamics of a dividing wall column. In: S. Pierucci, F. Manenti, G. L. Bozzano, D. Manca (Eds.), 30th European Symposium on Computer Aided Process Engineering. Vol. 48 of Computer Aided Chemical Engineering. Elsevier.

V. Tjeng, K. Xiao, R. Tedrake, 2017. Evaluating robustness of neural networks with mixed integer programming. arXiv preprint arXiv:1711.07356.