Provable Fairness for Natural Language Models using Neural Network Formal Verification

Proposal for Alexa Fairness in AI Amazon Research Award (Winter 2022)

PI: Stanley Bak, Assistant Professor, Computer Science, Stony Brook University
Co-PI: Steven Skiena, Empire Innovation Professor, Computer Science, Stony Brook University
Cash Funding Needed: $80K
AWS Promotional Credits Needed: $10K

Abstract

Machine learning models are increasingly deployed for important decision-making tasks, making it important to verify that they do not contain gender or racial biases picked up from training data. Typical approaches to achieve fairness revolve around efforts to clean or curate training data, with post-hoc statistical evaluation of the fairness of the model on evaluation data.

In contrast, we propose techniques to *prove* the fairness properties using recently developed formal methods that verify properties of neural network models. Beyond the strength of guarantee implied by a formal proof, our methods have the advantage that we do not need explicit training or evaluation data (which is often proprietary) in order to analyze a given trained model. The research focus is on extending formal verification approaches to long short-term memory (LSTM) networks, as well as a systematic classification of formal fairness specifications that can be proven about a given network.

Keywords: Neural Network Verification, Provable Fairness, LSTM

Introduction

Formal approaches to neural network verification have been developed over the past few years [15, 22, 13, 10] that can in certain cases prove properties about all possible executions of a specific network. Existing work has focused on networks used in safety-critical control algorithms and robotics [11, 24], or guaranteed robustness for visual perception networks [20]. We propose to extend such methods—specifically those based on star set overapproximations [22]—to network architectures commonly used in natural language processing (NLP), such as gated recurrent unit (GRU) networks and long short-term memory (LSTM) networks [25]. In addition, we plan to systematically explore classes of fairness specifications that can be evaluated both statistically as well as using the developed formal approaches. If successful, the project will extend highly-accurate formal verification methods to support GRU and LSTM cells.

NLP networks and enable the possibility of evaluating a given network with respect to different definitions of fairness. Further, such fairness definitions could be embedded into the training process help create fair NLP systems, similar to methods for formal verification of perception systems have been used to create more robust vision systems.

Prior Work on LSTM and Transformer Verification. Many approaches have emerged in recent years for neural network verification, and the PI helps run an annual competition to evaluate method applicability and scalability [3]. Despite progress, less work has been done on generalizing analysis methods to architectures

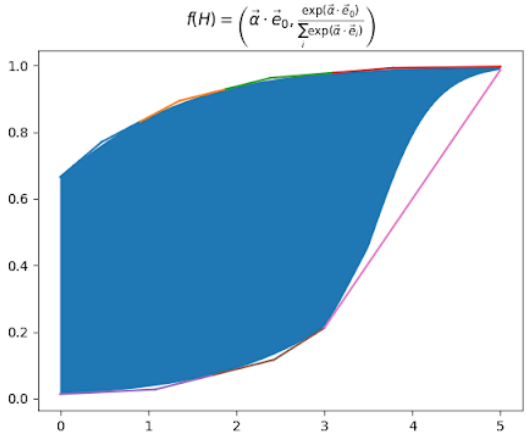

Figure 1: The star set approach can provide tight bounds on nonlinear layers such as softmax. The blue set is the real possible outputs of a neuron, projected onto the input (x axis) and output (y axis) of a single neuron.

and layers using in LSTM, due to complexity and nonlinearity. Some coarse analysis methods exist for recurrent and LTSM neural networks, such as those based on interval bounds [12], single upper and single lower linear function bounds [14, 18], bounding planes [7], polyhedral abstractions from samples [17], and multi-norm zonotopes [6]. Our approach, in contrast is based on efficient geometric data structures called linear star sets [8], which have been used by the PI in formal verification methods for cyber-physical systems [2, 5] and also extended to ReLU neural networks [21, 4, 1]. The advantage of this method compared with the existing approaches is the potential for higher accuracy—arbitrary linear constraints can be used to construct tight bounds on the nonlinear layers.

In our initial experiments, such methods are essential for creating tight bounds for softmax and self-attention layers. A figure of a prototype implementation showing linear bounds of a softmax layer is shown in Figure 1, where the true nonconvex output is in blue, and the outer convex linear bounds using the star set representation are shown. Based on this promising progress, we plan to integrate the approach into an open-source tool for proving fairness properties in LSTM networks, such as the PIs `nnenum` tool[1] which is available online[1] but currently only verifies ReLU networks.

Prior Work on Proving Fairness. A limited amount of existing research also exists on provable fairness. One framework focuses on proving *dependency fairness* [23, 9], which strives to prove the outputs are not affected by certain input features. This method is based on forward and a backward static analysis as well as input feature partitioning. The star set approach, in contrast, performs overapproximation and partitioning based on the nonconvexity in the activation functions, and so is likely to have improved scalability to larger networks or input spaces.

Another recent approach [16] focuses on *individual fairness*, which essentially means that similar individuals get similar treatments. In this case, the verification approach uses mixed integer linear programming (MILP), which can be as accurate as star sets, but in our experience with ReLU networks is often significantly less scalable in terms of network size and analysis time.

[1] https://github.com/stanleybak/nnenum

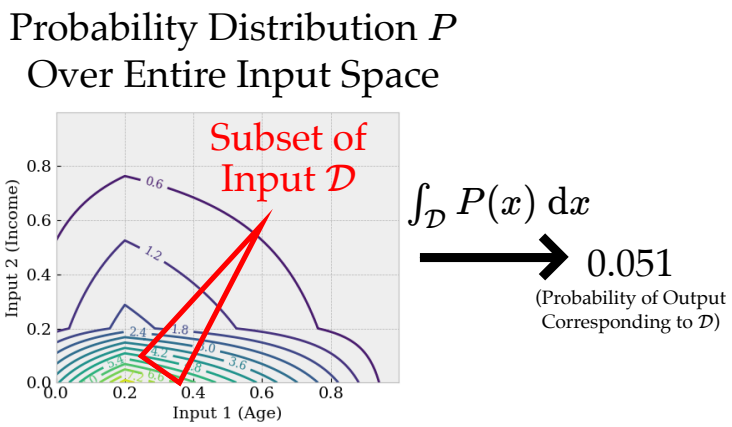

Figure 2: Each partition Θ of the network input space is used as the domain for integration over a given input probability density function P, calculating a probability of the input space with a fixed output classification. Repeating and summing over all partitions, we can evaluate fairness metrics for the network.

Methods

Our technical approach to verification is based on set-based execution of neural networks, extended to include probability distributions. Given a set of possible inputs, we propagate the set through the network to see the range of the network, the possible outputs. This process requires choosing representations for a set of states, with operations defined how the set transforms based on each layer in the network. In our proposed approach, we will use the linear star set representation [8, 19] to represent sets of states, which we simply call star sets. A star set is an affine trasnformation of a half-space polytope, where the affine transformation and polytope terms are kept separate. A star set Θ defines a set of states as $\Theta = \{x \in \mathbb{R}^n \mid \exists \alpha \in \mathbb{R}^m,\ x = V\alpha + c \wedge C\alpha \leq d\}$. This representation is useful for neural network verification because operations such as affine transformation, intersection, and optimization are accurate and efficient even in high-dimensional sets, corresponding to layers in a neural network with a large number of neurons. At the end of analysis, the entire input set is partitioned into a number of star sets, where each star set Θ maps a half-space polytope in the domain $\mathcal{D}$ (defined by $C\alpha \leq d$), to outputs with an identical label.

We plan to extend existing set propagation methods with star sets to work with layers in LSTM networks, and then using the result to reason over how probability distributions can be propagated through the network. Given a subset of the input space represented as a polytope in the domain $\mathcal{D}$ with a corresponding single output label, we can compute the probability density corresponding to that label integrating over the input domain, $\int_{\mathcal{D}} P(x)\ \mathrm{d}x$, where the probability density function $P(x)$ is defined over the entire input space. Repeating this over the entire input domain (all the star set partitions), we can compute the probability of each output over the entire domain of possible inputs. The process is illustrated in Figure 2 on a hypothetical 2-d input distribution, where a single partition over the input space given as a polytope $\mathcal{D}$ is used to compute a probability of a specific output.

The definition of fairness we strive to prove is then based on the calculated probability distributions. We will consider the following types of fairness challenges that can be verified using formal methods:

- **Classification with explicit input labels.** Here one of the input fields for each record explicitly encodes a sensitive property, such as race, gender, or age. Fairness dictates that a given model M performs "the same" for all possible values of these sensitive fields.
- **Classification with differential input distributions.** Here sensitive information is not explicitly

passed to the model but may be inferable from other input variables (e.g. the zip code of an African-American neighborhood). In addition to a given model M we are given input probability distributions for two or more distinct groups. Fairness dictates that M performs "the same" over records drawn from these distinct input distributions.

Every binary decision model partitions the space of possible inputs into a collection of geometric polytopes, such that all points within any particular polytope $\mathcal{D}$ are labeled homogeneously as all positive or all negative. The fairness properties which we are concerned with revolve around showing that these geometric regions are comparable for two distinct labels or groups. There are several possible provable properties depending upon how we define that M performs "the same" for different groups, including:

- **Volumetric identity.** Here we say that M is provably fair when the positive regions are identical for different groups A and B, ignoring the input variable explicitly encoding the identity of group A and B. Volumetric magnitude – Here we say M is provably fair when the volume of the positive regions are comparable for different groups A and B. The symmetric difference of the positive regions of A and B may be greater than zero, but the difference of the size of the input volumes is provably bounded.
- **Probability-weighted magnitude.** Here we say that M is provably fair with respect to input distributions A and B when the weighted volumes of the positive regions are the same for A and B, or that the difference of these weighted volumes is provably bounded. The weighted volume of polytope $\mathcal{D}$ is defined by the integral of the probability of each point in $\mathcal{D}$.

Expected Results

We anticipate open-source code releases with the work, along with publications related to the basic research and associated presentations. The code will be continuously released on github and publications will likely become available during the second half of the project, in Spring 2023. At this point we anticipate 2-3 publications related to the basic research and tool implementation of the developed methods. Presentations and publications will be made available online for widest dissemination of research results. In addition to the direct products of the work, we anticipate some of the fairness specifications can be interested into the neural network verification competition run by the PI [3]. In this way, the work can leverage research effort by other groups around the world to develop methods to address fairness concerns in NLP systems.

Funds Needed

For the project we plan to fund two graduate students as research assistants. Each of the three semesters (including summer) costs $11K, with $2826 tuition for Spring and Fall. The total direct costs for salary is 2 students * 3 semesters * $11K + 2 students * 2 semesters * $2826 = $77304. We also plan to allocate $2696 for domestic travel to conferences related to formal methods and verification such as CAV or NeurIPS.

For Amazon promotional credits, we are requesting $10K. In past research, our group has used EC2 to evaluate verification approaches using EC2 instances both with powerful CPUs and GPUs, and we anticipate this project will require similar resources for development and evaluation for fairness verification techniques.

Additional Information

Our proposed improvements to verification methods are targeted to layers and architectures used for NLP systems. We plan to integrate research developments into the PI's open source verification tool, `nnenum`, which currently only supports analysis of ReLU networks without probability distributions over the inputs.

References

[1] S. Bak. nnenum: Verification of ReLU neural networks with optimized abstraction refinement. In *NASA Formal Methods Symposium*, pages 19–36. Springer, 2021.

[2] S. Bak and P. S. Duggirala. Hylaa: A tool for computing simulation-equivalent reachability for linear systems. In *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control*. ACM, 2017.

[3] S. Bak, C. Liu, and T. T. Johnson. The second international verification of neural networks competition (VNN-COMP 2021): Summary and results. *CoRR*, abs/2109.00498, 2021.

[4] S. Bak, H.-D. Tran, K. Hobbs, and T. T. Johnson. Improved geometric path enumeration for verifying ReLU neural networks. In *32nd International Conference on Computer Aided Verification*. Springer, 2020.

[5] S. Bak, H.-D. Tran, and T. T. Johnson. Numerical verification of affine systems with up to a billion dimensions. In *Proceedings of the 22Nd ACM International Conference on Hybrid Systems: Computation and Control*, HSCC '19, pages 23–32, New York, NY, USA, 2019. ACM.

[6] G. Bonaert, D. I. Dimitrov, M. Baader, and M. Vechev. Fast and precise certification of transformers. In *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*, pages 466–481, 2021.

[7] T. Du, S. Ji, L. Shen, Y. Zhang, J. Li, J. Shi, C. Fang, J. Yin, R. Beyah, and T. Wang. Cert-rnn: Towards certifying the robustness of recurrent neural networks. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 516–534, 2021.

[8] P. S. Duggirala and M. Viswanathan. Parsimonious, simulation based verification of linear systems. In *International Conference on Computer Aided Verification*, pages 477–494. Springer, 2016.

[9] S. Galhotra, Y. Brun, and A. Meliou. Fairness testing: testing software for discrimination. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, pages 498–510, 2017.

[10] T. Gehr, M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, and M. Vechev. Ai2: Safety and robustness certification of neural networks with abstract interpretation. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE, 2018.

[11] R. Ivanov, J. Weimer, R. Alur, G. J. Pappas, and I. Lee. Verisig: verifying safety properties of hybrid systems with neural network controllers. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, pages 169–178, 2019.

[12] R. Jia, A. Raghunathan, K. Göksel, and P. Liang. Certified robustness to adversarial word substitutions. *arXiv preprint arXiv:1909.00986*, 2019.

[13] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*. Springer, 2017.

[14] C.-Y. Ko, Z. Lyu, L. Weng, L. Daniel, N. Wong, and D. Lin. Popqorn: Quantifying robustness of recurrent neural networks. In *International Conference on Machine Learning*, pages 3468–3477. PMLR, 2019.

[15] C. Liu, T. Arnon, C. Lazarus, C. Strong, C. Barrett, and M. J. Kochenderfer. Algorithms for verifying deep neural networks. *arXiv preprint arXiv:1903.06758*, 2019.

[16] A. Ruoss, M. Balunović, M. Fischer, and M. Vechev. Learning certified individually fair representations. *arXiv preprint arXiv:2002.10312*, 2020.

[17] W. Ryou, J. Chen, M. Balunovic, G. Singh, A. Dan, and M. Vechev. Scalable polyhedral verification of recurrent neural networks. In *International Conference on Computer Aided Verification*, pages 225–248. Springer, 2021.

[18] Z. Shi, H. Zhang, K.-W. Chang, M. Huang, and C.-J. Hsieh. Robustness verification for transformers. *arXiv preprint arXiv:2002.06622*, 2020.

[19] H.-D. Tran, S. Bak, W. Xiang, and T. T. Johnson. Verification of deep convolutional neural networks using imagestars. In *Proceedings of the 32nd International Conference on Computer Aided Verification*. Springer, 2020.

[20] H.-D. Tran, F. Cai, M. L. Diego, P. Musau, T. T. Johnson, and X. Koutsoukos. Safety verification of cyber-physical systems with reinforcement learning control. *ACM Transactions on Embedded Computing Systems (TECS)*, 18(5s):1–22, 2019.

[21] H.-D. Tran, D. M. Lopez, P. Musau, X. Yang, L. V. Nguyen, W. Xiang, and T. T. Johnson. Star-based reachability analysis of deep neural networks. In *International Symposium on Formal Methods*, pages 670–686. Springer, 2019.

[22] H.-D. Tran, W. Xiang, and T. T. Johnson. Verification approaches for learning-enabled autonomous cyber-physical systems. *IEEE Design & Test*, 2020.

[23] C. Urban, M. Christakis, V. Wüstholz, and F. Zhang. Perfectly parallel fairness certification of neural networks. *Proceedings of the ACM on Programming Languages*, 4(OOPSLA):1–30, 2020.

[24] W. Xiang, H.-D. Tran, X. Yang, and T. T. Johnson. Reachable set estimation for neural network control systems: A simulation-guided approach. *IEEE Transactions on Neural Networks and Learning Systems*, 32(5):1821–1830, 2020.

[25] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015.

Stanley Bak

stanley.bak@stonybrook.edu • www.stanleybak.com

Stanley Bak is an assistant professor in the Department of Computer Science at Stony Brook University investigating the verification of **autonomy**, **cyber-physical systems**, and **neural networks**.

Education

- **Doctor of Philosophy in Computer Science** — **May 2013**
  University of Illinois at Urbana-Champaign — *Champaign, IL*
  Dissertation Title: "Verifiable COTS-based Cyber-Physical Systems"
  Advisors: Marco Caccamo and Lui Sha

Positions

- **Assistant Professor, Department of Computer Science** — **September 2020-Present**
  Stony Brook University — *Stony Brook, NY*

  Investigated formal verification methods for autonomy, cyber-physical systems, and neural networks. Wrote and received competitive grant proposals from government agencies and ran student research group while teaching courses at the graduate and undergraduate levels.

Relevant Publications

Publication Metrics: According to Google Scholar, Stanley Bak has 2040 citations and an h-index 26 as of January 16, 2022.

- "The Second International Verification of Neural Networks Competition (VNN-COMP 2021): Summary and Results", S. Bak, C. Liu, and T. T. johnson, 4th Workshop on Formal Methods for ML-Enabled Autonomous Systems (FoMLAS) (VNNCOMP 2021)
- "nnenum: Verification of ReLU Neural Networks with Optimized Abstraction Refinement", S. Bak, 13th NASA Formal Methods Symposium (NFM 2021), 36% acceptance rate
- "Verification of Neural Network Compression of ACAS Xu Lookup Tables with Star Set Reachability", D. Lopez, T. Johnson; H.D. Tran, S. Bak, X. Chen, and K. Hobbs, AIAA Scitech Forum (SCITECH 2021)
- "Improved Geometric Path Enumeration for Verifying ReLU Neural Networks", S. Bak, H.D Tran, K. Hobbs and T. T. Johnson, 32nd International Conference on Computer-Aided Verification (CAV 2020), 27% acceptance rate
- "Verification of Deep Convolutional Neural Networks Using ImageStars", H.D Tran, S. Bak, W. Xiang and T. T. Johnson, 32nd International Conference on Computer-Aided Verification (CAV 2020), 27% acceptance rate
- "NNV: The Neural Network Verification Tool for Deep Neural Networks and Learning-Enabled Cyber-Physical Systems", H. Tran, X. Yang, D. Lopez, P. Masau, L. Nguyen, W. Xiang, S. Bak and T. Johnson, 32nd International Conference on Computer-Aided Verification (CAV 2020), 27% acceptance rate

Steven Skiena

http://www.cs.stonybrook.edu/∼skiena

(a) Professional Preparation

University of Virginia, Computer Science, B.S., 1983.
University of Illinois, Computer Science, M.S., 1985.
University of Illinois, Computer Science, Ph.D, 1988.

(b) Appointments

2018–	Empire Innovation Professor and Director of the AI Institute
2009–	Distinguished Teaching Professor, Stony Brook University.
2009–2015	Chief Scientist and co-founder, General Sentiment LLC
2001–2009	Professor of Computer Science, Stony Brook University.
1994–2001	Associate Professor of Computer Science, Stony Brook University.
1988–1994	Assistant Professor of Computer Science, Stony Brook University.

(c) Publications

1. B. Perozzi, R. al-Rfou, and S. Skiena. DeepWalk: Online Learning of Social Representations *20th ACM SIGKDD Conf. Knowledge Discovery and Data Mining* (KDD 2014), New York NY, August 2014.

2. R. Al-Rfou, V. Kulkarni, B. Perozzi, and S. Skiena, Polyglot-NER: Massively Multilingual Named Entity Recognition, *SIAM Int. Conf. on Data Mining (SDM 2015)*, Vancouver, BC, April 30-May 2, 2015.

3. S. Skiena, *The Data Science Design Manual* Springer International Publishing AG, Switzerland 2017. Russian translation to be published by BHV Publishing House, Russia. Korean translation to be published by Agency-One.

4. A. Kim, C. Pethe, and S. Skiena. What time is it? Temporal Analysis of Novels, Proc. Empirical Methods in Natural Language Processing (EMNLP), 2020.

5. C. Pethe, A. Kim and S. Skiena, Chapter Captor: Text Segmentation in Novels, Proc. Empirical Methods in Natural Language Processing (EMNLP), 2020.

6. S. Skiena, *The Algorithm Design Manual*, Springer-Verlag, New York, third edition, 2021.

Previously Funded Project Summary

Neither the PI and co-PI do not have previous funded projects with Amazon within the past 3 years.