The Case for Scalable Quantitative Neural Network Analysis*

Mara Downing
University of California Santa Barbara
USA
maradowning@cs.ucsb.edu

Tevfik Bultan
University of California Santa Barbara
USA
bultan@cs.ucsb.edu

ABSTRACT

Neural networks are an increasingly common tool for solving problems that require complex analysis and pattern matching, such as identifying stop signs in a self driving car or processing medical imagery during diagnosis. Accordingly, verification of neural networks for safety and correctness is of great importance, as mispredictions can have catastrophic results in safety critical domains. As neural networks are known to be sensitive to small changes in input, leading to vulnerabilities and adversarial attacks, analyzing the robustness of networks to small changes in input is a key piece of evaluating their safety and correctness. However, there are many real-world scenarios where the requirements of robustness are not clear cut, and it is crucial to develop measures that assess the level of robustness of a given neural network model and compare levels of robustness across different models, rather than using a binary characterization such as robust vs. not robust.

We believe there is great need for developing scalable quantitative robustness verification techniques for neural networks. Formal verification techniques can provide guarantees of correctness, but most existing approaches do not provide quantitative robustness measures and are not effective in analyzing real-world network sizes. On the other hand, sampling-based quantitative robustness is not hindered much by the size of networks but cannot provide sound guarantees of quantitative results. We believe more research is needed to address the limitations of both symbolic and sampling-based verification approaches and create sound, scalable techniques for quantitative robustness verification of neural networks.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; **Artificial intelligence**; • **Software and its engineering** → **Formal software verification**; *Empirical software validation.*

KEYWORDS

Neural Network Verification, Quantitative Verification, Safety-Critical Systems

ACM Reference Format:
Mara Downing and Tevfik Bultan. 2023. The Case for Scalable Quantitative Neural Network Analysis. In *Proceedings of the 1st International Workshop on Dependability and Trustworthiness of Safety-Critical Systems with Machine Learned Components (SE4SafeML '23), December 4, 2023, San Francisco, CA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3617574.3617862

*This material is based on research supported by NSF under Award #2124039

SE4SafeML '23, December 4, 2023, San Francisco, CA, USA

ACM ISBN 979-8-4007-0379-9/23/12.
https://doi.org/10.1145/3617574.3617862

1 INTRODUCTION

Machine learning techniques based on neural networks have revolutionized computer vision [9, 22, 28], speech recognition [1, 18], and natural language processing [26]. The technological revolution caused by advancements in machine learning has had a significant impact on society already, and its impact continues to broaden as neural network techniques are rapidly being adopted in a wide range of domains. Increasing deployment of neural network techniques in safety-critical and socially sensitive areas (e.g., self-driving cars [5, 8], robotics [2, 11], computer security [29], criminal justice [23], and medical diagnosis [30]) has created an urgent need to address dependability and safety of neural networks in a systematic and principled way.

Failures and unexpected behavior from these safety-critical AI systems are all too common, including notable examples such as the test of a military drone [33] that determined (via AI) that the best way to fulfill its objective was to kill the human in charge of its operations. More commonly in day-to-day life, the stories of various self driving car crashes fill newspapers as self-driving cars crash in ways that seem easy to avoid, such as running into stopped police cars as is reported in [19].

In some cases, like in [33], it is clear that an outcome should never occur. Traditional verification, which poses a constraint and asks the yes/no question of whether or not an outcome is possible, can handle these such cases. However, in domains such as self-driving cars it is common that avoiding all undesirable outcomes is not entirely possible, and in this case traditional verification will not suffice. It is not always possible to expect perfection—for example, it is intuitive that building a self-driving car that can never crash is unachievable. Thus, being able to produce a usable quantitative metric for the likelihood of crashes is a necessity.

A common method of verification for neural networks is robustness verification, where a correctly classified input and a perturbation radius around that input is given, and the verifier returns either (traditionally) whether or not a differently classified input exists in the radius, or (quantitatively) how many incorrectly classified inputs exist in that radius.

Quantitative robustness, as opposed to traditional robustness verification, makes it possible to have a meaningful comparison between the robustness of two neural networks in determining the best one for the problem at hand. If two networks both show an input and perturbation radius to be not robust with traditional verification, no more information is given to differentiate them. However, with quantitative robustness, it is possible to further compare of those two networks which one is **more** robust on the input

and perturbation in question. For a case where 100% robustness is not reasonably achievable, for example if determining whether there exists any perturbation to a stop sign that makes it unrecognizable, this ability to obtain levels of robustness allows for evaluation and comparison of networks that cannot be expected to have 100% robustness on an input and perturbation.

2 MOTIVATING EXAMPLES

Within this section we present a few cases from neural networks in the real world where quantitative analysis could help shed light on the likelihood and causes of undesirable outcomes, allowing for more understanding and also hopefully better analysis of safety-critical systems before release.

2.1 Natural Language Processing and Generation

ChatGPT [26] and similar tools have made great strides in language processing, but have also shown some notable and dangerous drawbacks. One of these is the creation and citation of fake sources, which has shown to have concerning consequences such as accusing people of crimes they could not have committed [38]. This is a clear case where traditional verification will not suffice. We already know a tool like ChatGPT can be incorrect or produce false citations, but for adequate analysis of the risk we need to know **how often** this can occur—which necessitates quantitative analysis. With analysis of the likelihood of incorrect claims, it would be possible to give a level of confidence for a result given by ChatGPT and help users approach answers with an understanding of how much those answers should be trusted. Analyzing the likelihood of false claims in different scenarios can also give insight into which conditions are most conducive to this issue, and thus allow for better training of future language models.

2.2 Self-Driving Cars

Many companies are now working on self-driving cars, for the dream that one day we could just tell our car where we want to go, sit back, and read a book while the car does the navigation for us. Safety, however, is a key concern not just for the passengers of these cars but for everyone else interacting with the roads—other drivers, bikers, pedestrians, and emergency responders [10].

The only way to be truly safe from a crash is to not move the car; there will always be situations possible where unexpected decisions made by other drivers can create an impossible-to-avoid crash. The measure of car safety thus cannot be whether or not it can crash, but rather how often it will crash, and how dangerous those crashes will be. Multiple articles [25, 31, 35] show Tesla's efforts to conceal and manipulate these numbers. In no case are they, or anyone else, arguing that they should have zero crashes—rather it is all about the numbers, the quantity of crashes per mile and how that compares to a human driver. This shows a need for quantitative verification—the likelihood of misclassifications and incorrect decisions by neural networks used in self-driving cars will help in estimating the likelihood of crashes and be able to identify likely failure patterns before human lives are on the line.

One example of a specific verification query would be to analyze the likelihood of incorrect lane identification in rain—we know that rain can cause incorrect predictions [36], and it would be unrealistic to expect that rain could never cause problems—with heavy enough rain, a human driver will also struggle with or be unable to correctly identify lane markers and street signs, so evaluating questions such as which network from a set of neural networks produces fewer misclassifications under harsh conditions will be more helpful than just declaring that all tested neural networks are not completely robust to rain-induced perturbations.

3 QUANTITATIVE ROBUSTNESS

Traditional robustness asks a yes/no question—does a misclassified input exist in a given perturbation region. However, a yes/no answer to a verification query about robustness does not give any information about how many of the perturbations change the output. For example in Fig. 1, both of these examples would be determined not robust by a traditional verifier. As both neural networks fail the robustness test, we cannot determine which one misclassifies fewer perturbed inputs. Alternatively, with quantitative verification the number of misclassified inputs is counted, and thus a distinction can be made. A network with a higher number of misclassified inputs in a given perturbation region is less robust (and, thus, more prone to adversarial attacks) than a network with fewer misclassified inputs in the same region.

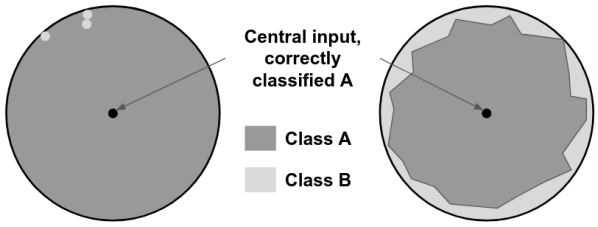

Figure 1: Image of two perturbation regions about an input, with different numbers of incorrectly classified inputs within the radius.

For a given neural network $\mathcal{N}$, an input $X_c = \langle x_0, \cdots, x_{n-1} \rangle$ (the center of the perturbation), and a perturbation limit $\Delta^{lim} = \langle \delta_0^{lim}, \cdots, \delta_{n-1}^{lim} \rangle$ (denoting the perturbation limit value per feature), the quantitative robustness measure $R(\mathcal{N}, X_c, \Delta^{lim})$ is as follows:

$$R(\mathcal{N}, X_c, \Delta^{lim}) = |S_{RobustSet}|/|S_{PerturbRegion}| \quad \text{where}$$

$$S_{RobustSet} = \{\tilde{X} \mid \arg\max \mathcal{N}(\tilde{X}) = \arg\max \mathcal{N}(X_c) \wedge x_i - \delta_i^{lim} \leq \tilde{x}_i \leq x_i + \delta_i^{lim}\}$$

$$S_{PerturbRegion} = \{\tilde{X} \mid x_i - \delta_i^{lim} \leq \tilde{x}_i \leq x_i + \delta_i^{lim}\}$$

In the definition above, $S_{PerturbRegion}$ denotes the set of all perturbed inputs within the perturbation region Δ^{lim}, and $S_{RobustSet}$ denotes the set of all perturbed inputs within the radius where the output of $\mathcal{N}$ does not change. Since all inputs in $S_{PerturbRegion}$ that are classified as expected are in $S_{RobustSet}$, we call remaining inputs potentially adversarial inputs, and we can define $S_{AdversarialSet}$ as follows: $S_{AdversarialSet} = S_{PerturbRegion} \setminus S_{RobustSet}$.

With traditional verification, if $|S_{AdversarialSet}| > 0$, then the network is determined not robust for that input and perturbation

radius. Quantitative robustness, alternatively, seeks to compute an exact or estimated value of $|S_{AdversarialSet}|$. Some common ways of achieving this are by counting $|S_{AdversarialSet}|$ or $|S_{RobustSet}|$ directly, or by estimating a ratio of $|S_{AdversarialSet}|/|S_{PerturbRegion}|$ via sampling a subset of the inputs in $S_{PerturbRegion}$.

It is possible to use traditional verification to produce a form of quantitative result—one way to leverage traditional verification for quantitative analysis is to compute minimum adversarial distortion (the closest incorrectly classified input to a given correctly classified input). By determining the radius at which the traditional robustness result changes from robust to not robust, it is possible to produce a sound lower bound on the minimum adversarial distortion, as is shown in [41]. However, this analysis misses the density of incorrectly classified inputs—for example, the two cases shown in Figure 1 would have very similar minimum adversarial distortions, despite their different quantitative robustness results.

4 QUANTITATIVE ROBUSTNESS VERIFICATION: CHALLENGES

There exist a number of verifiers for the robustness of full-precision networks [3, 6, 7, 13–17, 20, 21, 24, 27, 32, 37, 39–41]. However, the majority of these verifiers look at a traditional (non-quantitative) verification problem—asking whether or not a misclassified input exists in a perturbation region. A more in-depth analysis with quantitative verification asks instead how many misclassified inputs exist within the radius, to give a more detailed analysis of how vulnerable a network is to adversarial attacks—how likely it is to encounter one of these misclassifications. Within this quantitative realm, there exist a handful of full-precision quantitative verifiers [3, 27, 37, 40]. Of these verifiers, [27, 37] describe methods for quantitative verification based on symbolic analysis, which can produce reasonably precise results but are difficult to scale (Section 4.1). Alternatively, sampling-based quantitative verifiers [3, 40] are more scalable and are able to handle very large networks, but struggle to make a conclusive distinction between fully robust and almost fully robust regions (Section 4.2).

4.1 Symbolic Quantitative Verification

For symbolic quantitative verification [27, 37], a set of constraints is generated describing the behaviour of the neural network given an input and perturbation radius, and then a quantitative technique such as volume computation or model counting is used to determine how many solutions exist to the constraints. Volume computation produces results accurate in the real-valued domain, which is not exactly equivalent to the floating point domain, whereas model counting can provide exact results but is incredibly difficult for floating point constraints. Additionally, the the difficulty is compounded by the complexity of neural network constraints as shown in [37]. The difficulty of solving network constraints is also present in traditional symbolic robustness verifiers, but to a lesser extent as determining a count of satisfying solutions is a strictly more complex problem than computing satisfiability for a constraint. The symbolic approach, however, does allow for sound guarantees of full robustness rather than probabilistic guarantees as with sampling, and thus can differentiate between a fully robust region and a region with only a few incorrectly classified inputs.

4.2 Sampling-Based Quantitative Verification

Sampling-based quantitative verifiers [3, 40] do not have the same challenge as symbolic quantitative verifiers in that they don't have to handle complex constraints, instead they repeatedly run inputs through the network to achieve an estimate of the quantitative robustness. However, for a full precision network with multiple perturbed input features, there is no way to achieve an exact result in a reasonable amount of time—the number of distinct inputs to test explodes quickly with multiple features perturbed. Additionally, as is seen in [4] it takes many samples to achieve high confidence in a quantitative result via sampling. However, in many cases it is not necessary to achieve an exact or close-to-exact quantitative verification result, and a not so accurate result can be achieved reasonably quickly.

One of the key drawbacks to sampling for robustness, however, is that it struggles to distinguish between cases with few incorrectly classified inputs and zero, as without testing every possible input it is impossible to know for certain whether or not a lack of misclassified inputs found means there are none, or just few enough to have been missed by the sampling thus far. With L_∞-ball type perturbations, where each input feature (for example, each pixel) is perturbed by a small amount from its original value, the number of available inputs to test is easily beyond what can be tested in any reasonable amount of time.

5 FUTURE DIRECTIONS

Both sampling-based quantitative verification approaches and symbolic quantitative verification (which have been used effectively in the software engineering domain for program evaluation) show key strengths in the neural network domain, so one way to move forward and create a more effective quantitative verifier could be to combine symbolic and concrete/sampling-based verification into a hybrid verifier. Outside of the neural network domain, hybrid verification and testing tools have been investigated and have been successful [12, 34, 42].

The largest issue with sampling-based robustness verification, the inability to distinguish between 100% robustness and near-100% robustness, can be solved with symbolic quantitative verification to give useful network comparisons between networks with high, but not complete, robustness for a given input and perturbation radius. On the other hand, sampling can achieve usable approximate robustness for regions with lower robustness much faster than a symbolic approach. Thus, as the strengths of sampling and symbolic verification for the neural network domain are partially complimentary, finding a way to combine and balance the two is a promising direction of research in order to create usable verification tools for real-world neural networks.

Another direction to explore is extending and improving quantitative evaluation techniques for the types of constraints created by neural networks. This approach has had success in traditional network verification with Reluplex [20], which extends the simplex method for satisfiability checking to include rules for ReLU functions. Within the quantitative domain, either volume computation or model counting could be tailored to neural network constraints to allow for faster or more capable quantitative constraint solving and expand the capabilities of symbolic quantitative verifiers.

REFERENCES

[1] Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. 2014. Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on audio, speech, and language processing* 22, 10 (2014), 1533–1545.

[2] Ron Amadeo. 2023. *NYPD robocops: Hulking, 400-lb robots will start patrolling New York City.* https://arstechnica.com/gadgets/2023/04/nypd-robocops-hulking-400-lb-robots-will-start-patrolling-new-york-city/

[3] Teodora Baluta, Zheng Leong Chua, Kuldeep S Meel, and Prateek Saxena. 2021. Scalable quantitative verification for deep neural networks. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. IEEE, 312–323. https://doi.org/10.1109/icse43902.2021.00039

[4] Teodora Baluta, Shiqi Shen, Shweta Shinde, Kuldeep S Meel, and Prateek Saxena. 2019. Quantitative verification of neural networks and its security applications. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 1249–1264. https://doi.org/10.1145/3319535.3354245

[5] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. 2016. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316* (2016).

[6] Akhilan Boopathy, Tsui-Wei Weng, Pin-Yu Chen, Sijia Liu, and Luca Daniel. 2019. Cnn-cert: An efficient framework for certifying robustness of convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 3240–3247. https://doi.org/10.1609/aaai.v33i01.33013240

[7] Rudy Bunel, P Mudigonda, Ilker Turkaslan, P Torr, Jingyue Lu, and Pushmeet Kohli. 2020. Branch and bound for piecewise linear neural network verification. *Journal of Machine Learning Research* 21, 2020 (2020). https://doi.org/10.23919/acc.2018.8431048

[8] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. 2020. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11621–11631. https://doi.org/10.1109/cvpr42600.2020.01164

[9] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. 2017. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence* 40, 4 (2017), 834–848. https://doi.org/10.1109/tpami.2017.2699184

[10] Abené Clayton. 2023. *Fire chief warns against 'unleashing' self-driving taxis in San Francisco.* https://www.theguardian.com/us-news/2023/jun/23/self-driving-taxis-fire-chief-san-francisco

[11] Nicols Cruz and Javier Ruiz-del Solar. 2020. Closing the simulation-to-reality gap using generative neural networks: Training object detectors for soccer robotics in simulation as a case study. In *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8. https://doi.org/10.1109/ijcnn48605.2020.9207173

[12] Marko Dimjašević, Falk Howar, Kasper Luckow, and Zvonimir Rakamarić. 2018. Study of integrating random and symbolic testing for object-oriented software. In *Integrated Formal Methods: 14th International Conference, IFM 2018, Maynooth, Ireland, September 5-7, 2018, Proceedings 14*. Springer, 89–109. https://doi.org/10.1007/978-3-319-98938-9_6

[13] Souradeep Dutta, Susmit Jha, Sriram Sankaranarayanan, and Ashish Tiwari. 2018. Output range analysis for deep feedforward neural networks. In *NASA Formal Methods Symposium*. Springer, 121–138. https://doi.org/10.1007/978-3-319-77935-5_9

[14] Krishnamurthy Dvijotham, Robert Stanforth, Sven Gowal, Timothy A Mann, and Pushmeet Kohli. 2018. A Dual Approach to Scalable Verification of Deep Networks. In *UAI*, Vol. 1. 3.

[15] Ruediger Ehlers. 2017. Formal verification of piece-wise linear feed-forward neural networks. In *International Symposium on Automated Technology for Verification and Analysis*. Springer, 269–286. https://doi.org/10.1007/978-3-319-68167-2_19

[16] Timon Gehr, Matthew Mirman, Dana Drachsler-Cohen, Petar Tsankov, Swarat Chaudhuri, and Martin Vechev. 2018. Ai2: Safety and robustness certification of neural networks with abstract interpretation. In *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 3–18. https://doi.org/10.1109/sp.2018.00058

[17] Divya Gopinath, Kaiyuan Wang, Mengshi Zhang, Corina S Pasareanu, and Sarfraz Khurshid. 2018. Symbolic execution for deep neural networks. *arXiv preprint arXiv:1807.10439* (2018).

[18] Alex Graves and Navdeep Jaitly. 2014. Towards end-to-end speech recognition with recurrent neural networks. In *International conference on machine learning*. PMLR, 1764–1772.

[19] Chris Isidore. 2021. *Another Tesla reportedly using Autopilot hits a parked police car.* https://www.cnn.com/2021/08/30/business/tesla-crash-police-car/index.html

[20] Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. 2017. Reluplex: An efficient SMT solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*. Springer, 97–117. https://doi.org/10.1007/978-3-319-63387-9_5

[21] Guy Katz, Derek A Huang, Duligur Ibeling, Kyle Julian, Christopher Lazarus, Rachel Lim, Parth Shah, Shantanu Thakoor, Haoze Wu, Aleksandar Zeljić, et al. 2019. The marabou framework for verification and analysis of deep neural networks. In *International Conference on Computer Aided Verification*. Springer, 443–452. https://doi.org/10.1007/978-3-030-25540-4_26

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2017. Imagenet classification with deep convolutional neural networks. *Commun. ACM* 60, 6 (2017), 84–90.

[23] Yuan Y Liu, Min Yang, Malcolm Ramsay, Xiao S Li, and Jeremy W Coid. 2011. A comparison of logistic regression, classification and regression tree, and neural networks models in predicting violent re-offending. *Journal of Quantitative Criminology* 27 (2011), 547–573. https://doi.org/10.1007/s10940-011-9137-7

[24] Ravi Mangal, Aditya V Nori, and Alessandro Orso. 2019. Robustness of neural networks: A probabilistic and practical approach. In *2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*. IEEE, 93–96. https://doi.org/10.1109/icse-nier.2019.00032

[25] Erin Marquis. 2023. *Whistleblower Drops 100 Gigabytes of Tesla Secrets to German News Site: Report.* https://jalopnik.com/whistleblower-drops-100-gigabytes-of-tesla-secrets-to-g-1850476542

[26] OpenAI. 2022. *ChatGPT.* https://chat.openai.com

[27] Corina Păsăreanu, Hayes Converse, Antonio Filieri, and Divya Gopinath. 2020. On the probabilistic analysis of neural networks. In *Proceedings of the IEEE/ACM 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*. 5–8.

[28] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 779–788. https://doi.org/10.1109/cvpr.2016.91

[29] Joshua Saxe and Konstantin Berlin. 2015. Deep neural network based malware detection using two dimensional binary program features. In *2015 10th international conference on malicious and unwanted software (MALWARE)*. IEEE, 11–20. https://doi.org/10.1109/malware.2015.7413680

[30] Dinggang Shen, Guorong Wu, and Heung-Il Suk. 2017. Deep learning in medical image analysis. *Annual review of biomedical engineering* 19 (2017), 221–248.

[31] Faiz Siddiqui and Jeremy B. Merrill. 2023. *17 fatalities, 736 crashes: The shocking toll of Tesla's Autopilot.* https://www.washingtonpost.com/technology/2023/06/10/tesla-autopilot-crashes-elon-musk/

[32] Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. 2018. Boosting robustness certification of neural networks. In *International Conference on Learning Representations*.

[33] Guardian Staff. 2023. *US air force denies running simulation in which AI drone 'killed' operator.* https://www.theguardian.com/us-news/2023/jun/01/us-military-drone-ai-killed-operator-simulated-test

[34] Nick Stephens, John Grosen, Christopher Salls, Andrew Dutcher, Ruoyu Wang, Jacopo Corbetta, Yan Shoshitaishvili, Christopher Kruegel, and Giovanni Vigna. 2016. Driller: Augmenting Fuzzing Through Selective Symbolic Execution. https://doi.org/10.14722/ndss.2016.23368

[35] Brad Templeton. 2023. *Tesla Again Paints A Crash Data Story That Misleads Many Readers.* https://www.forbes.com/sites/bradtempleton/2023/04/26/tesla-again-paints-a-very-misleading-story-with-their-crash-data/?sh=7c533ab8feda

[36] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. 2018. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th international conference on software engineering*. 303–314.

[37] Muhammad Usman, Divya Gopinath, and Corina S Păsăreanu. 2021. QuantifyML: How Good is my Machine Learning Model? *arXiv preprint arXiv:2110.12588* (2021).

[38] Pranshu Verma and Will Oremus. 2023. ChatGPT invented a sexual harassment scandal and named a real law prof as the accused. *The Washington Post* (Apr 2023). Retrieved May 24, 2023 from https://www.washingtonpost.com/technology/2023/04/05/chatgpt-lies/

[39] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. 2018. Formal security analysis of neural networks using symbolic intervals. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*. 1599–1614.

[40] Stefan Webb, Tom Rainforth, Yee Whye Teh, and M Pawan Kumar. 2018. A statistical approach to assessing neural network robustness. *arXiv preprint arXiv:1811.07209* (2018).

[41] Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. 2018. Efficient neural network robustness certification with general activation functions. *arXiv preprint arXiv:1811.00866* (2018).

[42] Lei Zhao, Yue Duan, Heng Yin, and Jifeng Xuan. 2019. Send Hardest Problems My Way: Probabilistic Path Prioritization for Hybrid Fuzzing.. In *NDSS*. https://doi.org/10.14722/ndss.2019.23504

Received 2023-07-04; accepted 2023-08-10