The Reliability Stack for Machine Learning

Eric Wong (wongeric@mit.edu)

Machine learning (ML) systems have surpassed humans at a variety of tasks and benchmarks. However, deployed ML systems must go beyond these fixed settings: in practice, systems need to perform well in perpetually evolving environments. My research aims to develop the foundations of reliable systems in order to ensure their successful deployment in dynamic, real world conditions. To realize this goal, I study the "full stack" of reliability from the ground up: diagnosing problems, specifying changes, verifying properties, and training for robustness.

Reliable machine learning

State-of-the-art ML systems excel in settings that they were trained on, but can often struggle with unforeseen changes in the environment. Indeed, data-driven components for automobiles can perform poorly in bad weather [18]. Medical diagnosis models are significantly less accurate when deployed in new hospitals [14, 24]. National infrastructure such as smart grid systems can be unstable under severe weather or malicious interference [19, 20]. These examples illustrate a general failure of ML systems to adapt to changes. At the pinnacle of these shortcomings is their virtually non-existent robustness to worst-case inputs [22, 16]. These issues have driven a demand for more *reliable* ML systems: systems that perform consistently across a diverse array of setting. This goes beyond achieving better performance and is about building user trust—reliable systems have guarantees of safety/stability, allow for early detection of problems, and degrade gracefully to changes in the environment. My work has made progress towards these goals in the following areas:

1. **Robustness**—In the worst-case robustness setting, targeted changes to the input (attacks) can reduce accuracy of ML systems to 0%. There is a long history of researchers proposing new "robust" models (defenses) that were later shown to be completely ineffective against more advanced attacks [15]. One of the key contributions of my work was to bypass this arms race and develop *provable defenses* [1] that formally limit the worst-case performance of a system. My work established a principled methodology for creating differentiable guarantees that could be optimized during training. This line of work set the foundation for formal guarantees of *large-scale neural networks with general architectures* [2, 11].
2. **Specification**—How can we specify the conditions in which ML systems work (or not work)? Although robustness to noise is commonly studied, noise only capture a small subset of changes that occur in practice. To rigorously train and evaluate the reliability of systems to real-world effects, I developed new *perturbation sets* that go beyond synthetic noise and capture realistic changes [3, 8]. My work has also formulated a new class of *data-driven* perturbation sets [7], which enable specification of observable changes. We can now employ robust training to create the first models that are *adversarially robust to real-world effects learned from data*.
3. **Debugging**—ML systems that achieve top performance can also have hidden defects. To identify these issues, I developed debugging tools that enable the rapid *diagnosis of biases and spurious correlations* in ML systems [10]. These tools allow practitioners to proactively detect and resolve these flaws before deploying ML systems in the wild. To attain consistent and reliable debugging, my work has also shown how existing tools inject their own fundamental biases into the debugging process, and ways to mitigate these problems [12].

Beyond reliability, my work has identified how widely accepted guidelines from standard ML systems may no longer apply to *robust* ML systems. I showed how overfitting in the robust setting deviates from standard expectations [5, 6]. This led to surprising results for the field of worst-case robustness that challenged our basic understanding of generalization for robust models.

Researchers, scientists, and engineers will need ways to debug deficiencies in ML systems, specify and test the proper desired behavior, and ultimately re-deploy a corrected system. My vision is to develop the core infrastructure that enables this "reliability stack" across the spectrum of sciences. In the future, I aim to work with domain experts to understand their desiderata, and create the fundamental primitives that actualize new specifications and models for real-world robustness beyond standard ML benchmarks. I will also create new debuggability-centric tools and building blocks that enable us to both debug and create debuggable ML systems. In the remainder of this statement, I expand on my past research and future work.

1 Robustness

ML systems are known to make more errors when inputs deviate from what was seen during training. The extent of this non-robustness is embodied by adversarial examples, where small, targeted changes to the input can drastically harm the accuracy of a deep network. Indeed, the fact that a model can have vastly differing predictions when given two nearly identical inputs indicates that standard deep learning systems are fundamentally unreliable.

This led to an arms race between "attackers" and "defenders", where the attackers developed better techniques to generate adversarial examples, while defenders proposed methods to mitigate this problem. The burden of proof is heavily stacked against the defenders: an attacker needs to find only one adversarial example to break the model, while a defender must defend against all potential adversarial examples. The challenges of learning models that are robust in this *worst-case* setting became apparent when the vast majority of proposed defenses were shown to be fundamentally broken with stronger attacks [15], a trend that continues to occur [23].

One of my key contributions is a series of works [1, 2, 11] that established a principled methodology for training and verifying networks that are *provably* robust to adversarial attacks. This approach calculated analytical certificates that could guarantee robustness, side-stepping the need for empirical evaluations. In contrast to previous approaches from formal verification, which were combinatorial in runtime and limited to small networks, my work had the key insight to compute a tractable *upper bound* on the worst-case error rate [1]. Furthermore, this upper bound is differentiable and can be optimized to create deep networks with provably small worst-case error rates. With this approach, I created the first provably robust network with unbreakable robustness guarantees against adversarial examples[1]. Furthermore, this was the first formally verified *convolutional* architecture for standard computer vision benchmarks, which had previously been computationally infeasible to verify. In collaboration with researchers from Bosch, I used this framework in a real-world, fuel injection setting to create formal performance specifications of an ML system to sensor noise [4].

Verification approaches for deep learning typically require the network to be simple and small. However, in practice, deep networks are large and complex with a variety of modules and architectures. In order to train such networks with provable guarantees, I proposed a fully-modular framework for scalable robust training of *general computational graphs* [2]. I used this framework to create the first provably robust networks with popular deep learning components such as residual connections, pooling layers, and batch normalization. My later work built upon this approach to calculate even *tighter* upper bounds [11]. To this end, I have created the first GPU-accelerated linear programming solver for verifying the robustness of standard computer vision models. The solver scales convex programming well beyond the scope of typical commercial solvers, and produces the tightest bounds for the largest networks studied in formal verification.

The problem of creating ML systems with worst-case robustness is often framed as a robust optimization problem. This viewpoint is the backbone of most successful provable and empirical defenses. However, the landscape of robust optimization for deep learning is not well-understood. In my work, I discovered that adversarial training methods, a popular class of methods for robustly training deep networks, exhibit overfitting properties that are surprisingly different from standard training [5, 6]. These insights overturned commonly-accepted beliefs in adversarial training—by accounting for overfitting, older techniques originally seen as less effective were as competitive as newer approaches. I used these insights to drastically reduce robust training time from days to minutes [5], and isolated model selection as a major contributor to improvements in adversarial robustness [6].

2 Specification

At the core of robustness is the *perturbation set*—a formal specification of changes that an ML system should be robust to. Although synthetic noise is a popular choice, it does not cover all the natural changes that occur in the physical world. The study of reliability must go beyond noise to real-world changes, such as changing weather conditions or intentional graffiti tampering. However, research has predominantly focused on the noise model due to its simplicity and accessibility, leaving a significant disconnect between research and practice for reliable systems.

To bridge this gap, my work pioneered the development of perturbation sets that can formally specify structured changes beyond the simple noise model. I proposed the Wasserstein perturbation set [3], a

[1] Concurrent with [21]

mathematical model for adversarial examples that more naturally captures image transformations such as rotations and translations. To evaluate and train for robustness to Wasserstein adversarial examples, I developed a new algorithm for efficient projections onto Wasserstein balls. In later work, I oversaw the development of semantic perturbation sets for computer vision based on rendering engines [8] as well as generalizations of robust training to the union of multiple perturbation sets [9].

Despite our best efforts, certain real-world changes can be too complex for even humans to formalize. How can we characterize these challenging variations for robustness when we cannot formally describe them? I developed the first framework for *learning* perturbation sets from data, a general methodology for extracting data-driven specifications [7]. I used this framework to create data-based perturbation sets that capture real-world changes such as lighting, weather, and corruptions. These perturbation sets can generalize to new data, and enable comprehensive robustness evaluations beyond a fixed test set. Furthermore, my work bridged the gap between adversarial noise and real-world perturbations. Indeed, we can now use existing robust training algorithms that were previously seen as inapplicable to achieve robustness in the real-world.

3 Debugging

A system is only known to be as reliable as the axes used to measure its performance. If we do not know to check for a particular flaw a priori, an ML system that initially appears to be robust may severely deteriorate when deployed in new environments. An ideal reliable system can be quickly and easily debugged to find existing problems before deployment. This way, practitioners can take the corresponding steps to identify and correct the system.

But how can we diagnose unknown problems in ML systems? One approach is to hope that interpretability tools can reveal these flaws, but the sheer scale and complexity of modern ML systems can easily overwhelm a human. In my work, I took a different approach and instead modified ML systems to be *debuggable by design* [10]. I used tools from statistics and optimization to simplify deep networks in a way that allows humans to better understand the decision process without significantly compromising performance. With this framework, I showed how we can identify learned biases and spurious correlations in modern vision and language models, and discovered that "de-biased" language models were still biased. My work also enabled the creation of counterfactual inputs as well as explanations for mispredictions. These debugging modes provide usable information that exposes how the model makes predictions. In order to test and measure the degree to which model debuggability was enhanced, in this work, I designed new human experiments that importantly avoided human biases in the evaluation.

Our ability to debug ML systems is only as good as the tools used to inspect them. However, the inherent biases in ML systems can also disrupt these same tools. In a study on the impact of missing features, I demonstrated how widely-used ML systems can suffer from skewed and incorrect predictions when features are missing [12]. This has immediate consequences for popular interpretability tools that toggle features on and off. I showed that applying such tools to biased systems produces explanations that are no better than random explanations. These insights demonstrate how certain models are inherently more debuggable due to their internal biases. Our understanding of internal biases in ML systems have implications beyond debugging as well: in recent work, I used this knowledge to significantly improve robustness against real-world attacks on computer vision systems [13].

4 Future work

The study of reliability for ML systems is a multi-step cycle, where developers continuously identify issues, update specifications, and correct behaviors of ML systems. While most work in robustness considers these individual steps in isolation, I aim to build both the core primitives and the connecting bridges that make this entire "reliability stack" a reality. These foundations for reliable ML will need to be flexible and general, capable of adapting to a diverse array of applications and conditions. A longer term goal is to move towards an *automatic* pipeline that minimizes the need for human intervention throughout the process. Finally, I plan to use new techniques and methods for debugging models to more broadly understand the patterns that ML systems learn and how they make decisions.

Capturing real-world changes. The vast majority of work in robustness focuses on mathematically defined changes or generated datasets. In contrast, my recent work has only scratched the surface for

learning arbitrary specifications from data, with many avenues for future work. A natural direction is to explore the domain of data-driven specification beyond static images—what specifications can we learn from text, audio, or video data? Different domains naturally capture different variations, and present new modeling and robustness challenges for reliable ML. What are the proper generative models for capturing variations in different domains, and how can we learn them? What kinds of specifications do domain experts need, and can we adapt and scale traditional robustness guarantees to these settings? I hope to collaborate with domain experts and use my broad expertise in verification to enable reliability specifications for all types of ML systems.

Behind the algorithmic and modeling challenges is the *data* used to learn real-world variations in the first place. While ideally we would have access to observations that are labeled with the corresponding changes, this may be too expensive to collect. Where can we source data that captures natural variations, and how much data is needed? One potential source is to extract variations from large datasets, and utilize richer sources of *unlabeled* data. I plan to investigate new ways to data-mine observations of real-world changes from large datasets with minimal human supervision. What types of natural variations already exist in ML datasets? How can we extract or filter out these patterns, and are ML systems robust to these changes? How can we learn a model of naturally mined variations, and can these models be used to boost settings beyond robustness?

Debuggability-centric ML. The predominant paradigm for debugging ML systems is to apply these methods to fully trained, black-box models. Although general in applicability, these methods on their own are limited in the scope and type of insight that they can provide. This mismatch between the constraints of ML debugging tools and the complexity of modern ML systems restricts our ability to understand how a system makes predictions. Instead of directly using these methods to debug ML systems, which remains a challenging task, a different goal is to instead re-design ML systems from ground up with *debuggability as a core feature*.

What parts of the ML pipeline can we expose to the user and simplify without trading off performance? What are the basic building blocks that naturally reveal useful information to a human? My work on using sparse linear layers [10] is only one potential mechanism that can already improve debuggability without sacrificing performance. A natural next step is to re-design and sparsify initial processing layers, to reveal which input features are actually being used for the prediction process. However, debuggability-centric design can go beyond sparsifying existing system components. Incorporating intervention-friendly mechanisms such as causal structures or logical reasoning into our ML building blocks can unlock new ways to test and troubleshoot model behavior. I aim to continue developing new primitives for debuggability-centric ML systems: modules for processing complex patterns that can reveal usable information to the user.

Concluding Remark. Reliability has not always been a priority in machine learning, which has traditionally measured progress with static, in-distribution test sets. However, the ubiquitous use of ML has thrust the inconsistencies of modern systems beyond the test set into the limelight. The field of reliable ML is of considerable practical importance, as scientists and engineers increasingly adopt data-driven systems and become data scientists. I look forward to tackling these challenges as we work towards the next generation of systems that we can trust.

References

[1] Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5286–5295. PMLR, 2018.

[2] Eric Wong, Frank Schmidt, Jan Hendrik Metzen, and J. Zico Kolter. Scaling provable adversarial defenses. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[3] Eric Wong, Frank Schmidt, and Zico Kolter. Wasserstein adversarial examples via projected sinkhorn iterations. In *International Conference on Machine Learning*, pages 6808–6817. PMLR, 2019.

[4] Eric Wong, Tim Schneider, Joerg Schmitt, Frank R Schmidt, and J Zico Kolter. Neural network virtual sensors for fuel injection quantities with provable performance specifications. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1753–1758. IEEE, 2020.

[5] Eric Wong, Leslie Rice, and J. Zico Kolter. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=BJx040EFvH`.

[6] Leslie Rice, Eric Wong, and Zico Kolter. Overfitting in adversarially robust deep learning. In *International Conference on Machine Learning*, pages 8093–8104. PMLR, 2020.

[7] Eric Wong and J Zico Kolter. Learning perturbation sets for robust machine learning. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=MIDckA56aD`.

[8] Rahul Venkatesh, Eric Wong, and J Zico Kolter. Semantic adversarial robustness with differentiable ray-tracing.

[9] Pratyush Maini, Eric Wong, and Zico Kolter. Adversarial robustness against the union of multiple perturbation models. In *International Conference on Machine Learning*, pages 6640–6650. PMLR, 2020.

[10] Eric Wong, Shibani Santurkar, and Aleksander Madry. Leveraging sparse linear layers for debuggable deep networks. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 11205–11216. PMLR, 18–24 Jul 2021. URL `https://proceedings.mlr.press/v139/wong21b.html`.

[11] Shaoru Chen, Eric Wong, J Zico Kolter, and Mahyar Fazlyab. Deepsplit: Scalable verification of deep neural networks via operator splitting. *arXiv preprint arXiv:2106.09117*, 2021.

[12] Saachi Jain, Hadi Salman, Eric Wong, and Aleksander Madry. Missingness bias in model debugging. 2021.

[13] Hadi Salman, Saachi Jain, Eric Wong, and Aleksander Mądry. Certified patch robustness via smoothed vision transformers. *arXiv preprint arXiv:2110.07719*, 2021.

[14] Ehab A AlBadawy, Ashirbani Saha, and Maciej A Mazurowski. Deep learning for segmentation of brain tumors: Impact of cross-institutional training and testing. *Medical physics*, 45(3):1150–1158, 2018.

[15] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International conference on machine learning*, pages 274–283. PMLR, 2018.

[16] Battista Biggio, Igino Corona, Blaine Nelson, Benjamin IP Rubinstein, Davide Maiorca, Giorgio Fumera, Giorgio Giacinto, and Fabio Roli. Security evaluation of support vector machines in adversarial environments. In *Support Vector Machines Applications*, pages 105–153. Springer, 2014.

[17] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Brandon Tran, and Aleksander Madry. Adversarial robustness as a prior for learned representations. *arXiv preprint arXiv:1906.00945*, 2019.

[18] Modar Horani and Osamah Rawashdeh. A framework for vision-based lane line detection in adverse weather conditions using vehicle-to-infrastructure (v2i) communication. Technical report, 2019.

[19] Tian Liu and Tao Shu. Adversarial false data injection attack against nonlinear ac state estimation with ann in smart grid. In *International Conference on Security and Privacy in Communication Systems*, pages 365–379. Springer, 2019.

[20] Olufemi A Omitaomu and Haoran Niu. Artificial intelligence techniques in smart grid: A survey. *Smart Cities*, 4(2):548–568, 2021.

[21] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. *arXiv preprint arXiv:1801.09344*, 2018.

[22] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[23] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. *arXiv preprint arXiv:2002.08347*, 2020.

[24] John R Zech, Marcus A Badgeley, Manway Liu, Anthony B Costa, Joseph J Titano, and Eric Karl Oermann. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: a cross-sectional study. *PLoS medicine*, 15(11):e1002683, 2018.