Disorderly notes for an AI-enthusiastic structural engineer:

Note 1. *To challenge a model is to respect it: the need for strict model evaluation in structural engineering-oriented ML applications*

Mohsen Zaker Esteghamati

Department of Civil and Environmental Engineering, Utah State University, Logan, Utah, USA

August 2026

Version 1.0

DOI: https://doi.org/10.5281/zenodo.22151519

1 Prelude

A common practice in evaluating ML models is to randomly split a dataset into training and test sets. If a model performs well on the test set and its performance is similar to that on the training set, then the model is "expected" to behave "nicely" in the future. I would like to argue that the way we test our ML models has significant implications for their deployment and their future performance on unseen data. In other words, models that cannot generalize well are often models that were never "challenged" to do so.

Models, like people, need a "serious" challenge to show their potential, their own "*misogi*[1]". Needless to say, we often subject them to mundane evaluation, justifying it by assuming that the resulting test set mimics the training data and is therefore a fair basis for comparison. However, the fine print of using randomized train/test sets is that they require "sufficiently large" datasets so that the empirical distributions of the random subsets still, with high probability, reflect the underlying population distribution (see the Glivenko–Cantelli theorem). These conditions are often violated in small engineering datasets, especially those involving failure. In failure-related datasets, which one can assume are the bread and butter of structural engineering research, "sufficiently large" is not merely about the total number of observations, as the observations

[1] Read *The Comfort Crisis* by Michael Easter

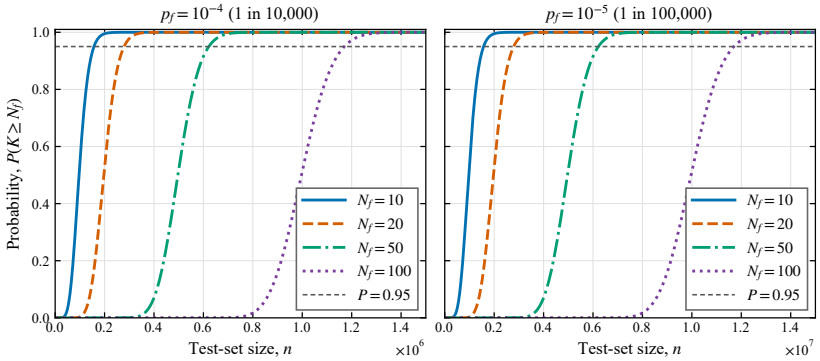

Figure 1: Probability of observing at least N_f failure cases as a function of test-set size for failure probabilities of $p_f = 10^{-4}$ and $p_f = 10^{-5}$. K is the number of failures observed in the test set, and N_f is the minimum number of failure cases one aims to observe.

of utmost interest are most often the ones we cannot afford to have in abundance [1, 2]. Here, even an apparently large dataset may remain painfully small when considering the tails of the distribution[2].

Suppose each test observation is an independent draw from a population in which the failure probability is $p_f = 10^{-4}$. If K denotes the number of failures in a test set of size n, then $K \sim \text{Binomial}(n, p_f)$. As shown in Figure 1, observing at least 10 failure cases with 95% probability, this requires approximately 160,000 test observations, whereas observing at least 100 failure cases with the same confidence requires approximately 1.2 million observations. For a rarer failure probability of one in 100,000, these requirements increase to 1.6 million and 12 million observations, respectively. Therefore, relying on randomized splits alone can quickly become uninformative, while assembling a sufficiently large test set may be infeasible [3]. Such challenges have precedents in the structural reliability field, where active-learning methods select informative points near the limit-state boundary, while importance sampling and subset simulation concentrate samples in important failure regions [3, 4].

"Data leakage" during the split is another issue that can be quite tricky to address and, as Kapoor and Narayanan warn, it is common and can lead to an overoptimistic view of model performance [5]. While certain cases of data leakage seem easy to avoid, our perception of leakage can significantly affect how we split our data into training and test sets. Consider the case where one aims to predict the cyclic behavior of a specimen under different loading scenarios. Should the data be separated by loading scenario, or should the

[2]The Dvoretzky–Kiefer–Wolfowitz inequality provides a bound on the worst-case distance between an empirical cumulative distribution function and the true cumulative distribution function

[3]A great read on this is the paper by Hanley and Lippman-Hand (1983) "If Nothing Goes Wrong, Is Everything All Right? Interpreting Zero Numerators"

time histories be aggregated and the responses randomly split into temporal windows, such that different "segments" of the same loading history are assigned to each set? What if the loading conditions produce substantially different cyclic responses? The way one answers these questions reflects the intended unit of generalization, which subsequently determines the appropriate split and may require grouped, blocked, or leave-one-specimen-out evaluation rather than observation-level random splitting [6, 7].

Failure-related datasets often exhibit class imbalance, and, as such, data augmentation techniques are often applied to better represent the minority class. Nevertheless, these techniques may be applied indiscriminately to the entire training data, producing "more and more of the same" through sophisticated means. Even when applied correctly, data augmentation is typically restricted to the training data and does not, by itself, make the test set more informative or challenging. In short, what the model needs is challenging cases to learn from and a separate, untouched set against which to be tested. However, as it seems, models often get only one of these—or neither— while the truly important cases may be poorly learned and just as poorly tested [8].

2 Illustrative example: model success is in the eye of the beholder

Figure 2 presents the results of testing two deep learning models (referred to here as Model A and Model B) on two test sets. These models are taken from our prior research on ML-based waveform quality assessment for ground-motion processing [9]. Test Set 1 was selected to be challenging because the underlying physical parameter (here earthquake source parameters, such as magnitude) differs significantly from that of the training data, whereas Test Set 2 is more similar to the training data. On Test Set 1, Model B appears slightly worse than Model A: its accuracy is 5 and 7 percentage points lower for H_1 and H_2 (i.e. two horizontal components of waveform), respectively, while its F1 score is one percentage point lower for both. Although Model B has higher recall, one may reasonably conclude from this test set that the two models perform similarly, or one may even favor Model A. When tested on Test Set 2, Model B outperforms Model A in accuracy by 9 and 12 percentage points and F1 score by 9 and 13 percentage points for H_1 and H_2, respectively. So, which one provides the more relevant measure of model success? Perhaps this is more of a philosophical question, and one that can be used to separate "half-full" from the "half-empty" crowd. I tend to fall into the half-empty camp and assume that, in a real-world deployment, the model's performance will be closer to Test Set 1, where the architectural advantage of Model B is far less clear. Often, research in the field will use Test Set 2 and attribute the observed difference to the model architecture, selecting Model B as the better-performing model[4].

[4]This is an informal illustration and a more rigorous comparison should report sample size for each test set to remove ambiguity. A one-percentage-point difference may simply reflect

This is why, after all, model success is in the eye of the beholder [10].

3 Concluding remarks

The first conclusion from this note is the need to move beyond "indiscriminate" randomized train/test splits to deliberate splits that preserve independence and align the test set with the conditions the model is expected to encounter during deployment. How does one recognize the important data regimes without first building the model? This test-design problem has, for the most part, always been a chicken-and-egg problem and is perhaps one reason why building useful ML models remains as much an art as a technical challenge. As with writing a good journal paper, one can start with a rough draft and continuously polish the model until it behaves nicely (or pleases Reviewer 2 in our analogy). The good news is that in structural engineering problems, we are often aware of the key parameters expected to influence the response and can ensure they are considered explicitly when constructing the training and test sets

My second thought is that if we would like to use ML models in structural engineering, perhaps we should adopt the philosophy embedded in our build-

sampling variability

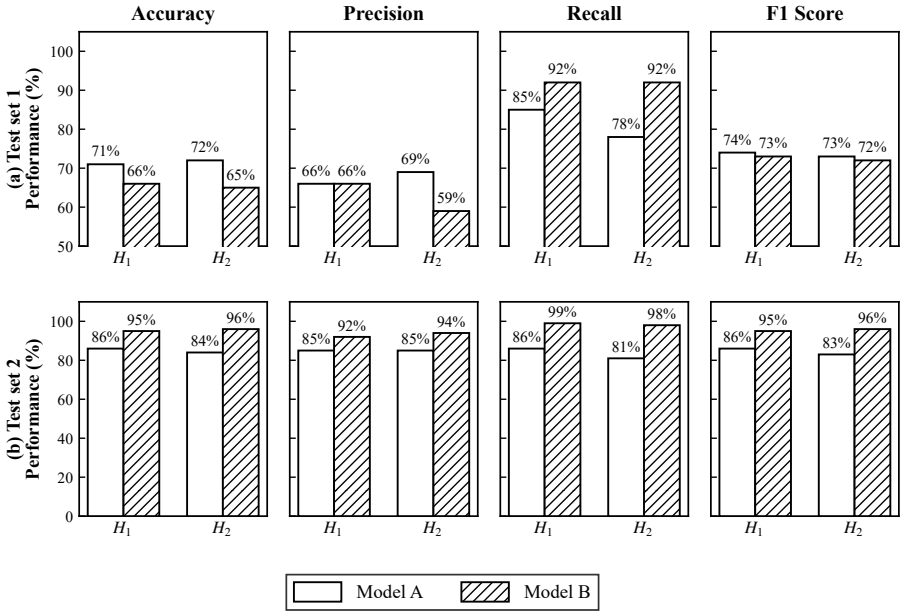

Figure 2: Comparison of the time-series-based and image-based models for H_1 and H_2 using accuracy, precision, recall, and F1 score on Test Set 1 and Test Set 2.

ing codes — that is, to be conservative (perhaps overly so) and evaluate our models on more demanding or even worst-case examples. While this approach may take some of the glamour away from models reporting $R^2 = 0.9998$, the models selected through such evaluation will probably be better suited for practical use. It is my honest opinion that, with current practices, more and more ML models in the field will "fail silently" [11, 12] during deployment, providing little to no benefit and remaining little more than "toy" models. This issue is not specific to our field and is related to what has been called "shortcut learning" [13] across different learning environments, i.e., the tendency of a learning system to discover an easy rule that succeeds on an ordinary examination but fails under a more demanding one. As a safety-critical field, instead of rejecting AI/ML, we may embrace it with our built-in obsession with "safety margins", perhaps through methods such as behavioral testing, stress testing, and adversarial examples[14, 15]. To me, that is a structural engineering-oriented way of using AI, and one that would benefit the field in the coming years.

References

[1] HaiYing Wang. Logistic regression for massive data with rare events. In *International conference on machine learning*, pages 9829–9836. PMLR, 2020.

[2] Rita P Ribeiro and Nuno Moniz. Imbalanced regression and extreme value prediction. *Machine Learning*, 109(9):1803–1835, 2020.

[3] Maliki Moustapha, Stefano Marelli, and Bruno Sudret. Active learning for structural reliability: Survey, general framework and benchmark. *Structural Safety*, 96:102174, 2022.

[4] Siu-Kui Au and James L Beck. Estimation of small failure probabilities in high dimensions by subset simulation. *Probabilistic engineering mechanics*, 16(4):263–277, 2001.

[5] Sayash Kapoor and Arvind Narayanan. Leakage and the reproducibility crisis in machine-learning-based science. *Patterns*, 4(9), 2023.

[6] Shachar Kaufman, Saharon Rosset, Claudia Perlich, and Ori Stitelman. Leakage in data mining: Formulation, detection, and avoidance. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(4):1–21, 2012.

[7] David R Roberts, Volker Bahn, Simone Ciuti, Mark S Boyce, Jane Elith, Gurutzeta Guillera-Arroita, Severin Hauenstein, José J Lahoz-Monfort, Boris Schröder, Wilfried Thuiller, et al. Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure. *Ecography*, 40(8):913–929, 2017.

[8] Gaël Varoquaux. Cross-validation failure: Small sample sizes lead to large error bars. *Neuroimage*, 180:68–77, 2018.

[9] Ali Namin, Albert Kottke, and Mohsen Zaker Esteghamati. Automating ground motion quality assessment using residual neural networks. In *Geo-Extreme 2025*, pages 75–85. 2025.

[10] David J Hand. Classifier technology and the illusion of progress. 2006.

[11] Stephan Rabanser, Stephan Günnemann, and Zachary Lipton. Failing loudly: An empirical study of methods for detecting dataset shift. *Advances in Neural Information Processing Systems*, 32, 2019.

[12] Mohsen Zaker Esteghamati, Brennan Bean, Henry V Burton, and MZ Naser. Beyond development: Challenges in deploying machine-learning models for structural engineering applications. *Journal of Structural Engineering*, 151(6):04025059, 2025.

[13] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[14] Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of nlp models with checklist. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 4902–4912, 2020.

[15] Eric Breck, Shanqing Cai, Eric Nielsen, Michael Salib, and D Sculley. The ml test score: A rubric for ml production readiness and technical debt reduction. In *2017 IEEE international conference on big data (big data)*, pages 1123–1132. IEEE, 2017.