Evaluating Machine Learning Models for Internet Network Security with Data Slices

Pamela Toman[*]
Palo Alto Networks
Santa Clara, CA 95054
`ptoman@paloaltonetworks.com`

Elisha Yadgaran
Palo Alto Networks
Santa Clara, CA 95054
`eyadgaran@paloaltonetworks.com`

Christina Papadimitriou
Palo Alto Networks
Santa Clara, CA 95054
`cpapadimitri@paloaltonetworks.com`

Aaron Isaksen
Palo Alto Networks
Santa Clara, CA 95054
`aisaksen@paloaltonetworks.com`

Matt Kraning
Palo Alto Networks
Santa Clara, CA 95054
`mkraning@paloaltonetworks.com`

Abstract

By using public data about the structure of the internet, practitioners can identify what assets organizations own on the internet, many of which are vulnerable to cybersecurity attacks. With current knowledge of what servers and assets are exposed to the internet, organizations are able to remediate vulnerabilities before they are exploited. As part of managing an AI/ML system for this "internet asset attribution" task, we make extensive formal use of "data slices", subsets of data that share particular properties. Data slices make managing models and datasets more repeatable and sustainable. Data slice evaluation lets us systematically manage regressions, user trust, experiment evaluation, and data space characterization.

1 Introduction

Operationalizing machine learning systems in cybersecurity is particularly difficult. In this paper we describe a data-centric approach where we use data subsets to evaluate the ML internet asset attribution problem, a subdomain of cybersecurity. In addition to significant variability of internet traffic, the high cost of both false positives and negatives, and the challenge of thorough model evaluation and lack of standardized test sets [12], the environment is actively adversarial. Furthermore, operators in the field are overwhelmed by volume; the accuracy of ML systems is limited by their ability to suppress false positives when most data is benign [1]. As in many domains, quality labeled data is hard to obtain, as privacy and safety concerns discourage disclosure of both benign and malicious samples. In general, it is far more critical to ML cybersecurity applications to use representative train and testing data than to focus on method [2].

We have developed a family of machine learning models that predicts whether components of the internet belong to particular organizations. The "attacker's eye view" of asset ownership these models

[*]Pamela Toman is the corresponding author. Elisha Yadgaran initially proposed using data slices to frame our team's multi-objective work, which has grown into the set of useful practices described here.

enable are key to computer network defense.[2] These models support organizations' attack surface management. We define attack surface [13, 5] as "the assets that an organization owns which are visible on the internet", and asset attribution means finding those assets for a particular organization. The model decides ownership given an internet-facing asset, like a website domain name, IP address, TLS/SSL certificate, and other public internet data. Our modeling and cybersecurity goal is to build a complete, current and accurate view of the assets on an organization's internet attack surface.

We use data subsets because we often find that we care about dimensions of our modeling task beyond strictly predictive attribution performance. In addition to developing a test set for overall model comparison, we formulate and maintain "data slices" that share some property. We use the term "data slices" because it relates closely to slice-based learning [3], where they define slices as "application-critical data subsets, specified programmatically by machine learning practitioners, for which we would like to improve model performance". By using data slices as part of our evaluation, we make limited and valuable labeled data maximally useful in a rigorous way.

Crafting data slices enables modelers to make assertions about multi-faceted reality. Instrumentation by "kinds of data" lets researchers know if data/feature/model interventions to address performance blindspots produce improvements. Data slice instrumentation also facilitates monitoring whether the model regresses on a data region over time. Data slices are useful for characterizing the dimensions on which two models with similar test set performance differ. Like feature explanations characterize models in feature space, data slices characterize models in data space.

2 Approach to data slice creation

A data slice is a strict subset of the entire distribution expected to be seen by the machine learning application. Any subset can be formulated into a slice and independent data slices may have overlapping segments. Data slices can contain labeled (either manually or automatically) [10] data for evaluation and supervised learning, or unlabeled data for reinforcement learning or unsupervised learning. In this paper we focus on data slices for evaluation. Our test set and loss function are related to each other, but not to the data slices used in evaluation.

Each data slice has important properties which should be stored as metadata with the data slice:

Name. A human-readable unique name for the data slice (e.g., `SpanishLanguageDomains`).

UUID. A globally unique identifier for referring to the data slice.

Purpose. The reason this data slice exists so others understand what it represents (e.g., "only internet domains with Spanish words, to determine how our algorithms function on Spanish").

Lineage. The immutable source data from which the data slice was extracted.

Time window. The period of time that the data slice covers.

Sampling details. The sampling method, random seeds, and other facts required to reproduce this data slice (e.g., `method=default, random_seed=1234, n_samples=5000`).

Data slices are typically generated with a sampling function and a data query that pulls data that shares some common trait. Common traits identified deductively by humans could be explicit (such as an explicit property and/or label) or implicit (such as a human-observable characteristic that cannot be extracted 1:1 from raw data, which must be iteratively uncovered through increasing familiarity with data). Directed autogeneration techniques can also be used to create data slices. These inductive data-driven techniques iteratively decompose a large dataset into data slices, such as along feature values or by clustering. Finally, we also create synthetic data slices explicitly engineered to always elicit a consistent label from a human labeler. Inductive, deductive, and synthetic approaches complement each other by addressing aspects that the other approaches may miss.

[2]Contrary to what one might expect, organizations rarely know or monitor even 90% of what they actually have exposed to the internet. Even worse, the unmonitored assets tend to be older and poorly managed, exhibiting more security vulnerabilities. With an up-to-date awareness of whether a newly visible-to-the-public website domain or other internet asset actually belongs to them, organizations can defend themselves against adversaries.

3 Discussion of usage in practice

In this section, we discuss data slices as applied to the network security use case, our gating techniques, data management, and ethical considerations.

Organizations rarely know all of their servers that face the public internet, and even more rarely closely monitor all those servers. However, any connection to the public internet is a potential vector for malware, negative publicity, and loss of user data. Organizations frequently mitigate their exposure through deploying software "agents" on servers, laptops, and other devices [11]. These agents monitor the behavior of the device. Centralized security software receives reports from the agents, and that software can take actions like quarantining misbehaving devices.

Unfortunately, even organizations with robust software agents tend to have "legacy" or "shadow" IT that is not under management, and software running on cloud providers that is not fully instrumented. Organizations also have assets that are configured dangerously in a way not reportable by the agent, and organizations rely on internet-facing devices that cannot run software agents (e.g., teleconference hardware cannot run agents but does run vulnerable software).

We use ML to find an "attacker's eye view" of the computer network using public data about the structure of the internet, like content, redirect records, and registration records. The asset attribution ML model finds assets that are vulnerable to the Meris botnet [7] or to being taken over for bitcoin mining [6], systems vulnerable to dangerous nation-state exploits like the NSA's EternalBlue [8], and hijacked or newly announced IP ranges potentially registered by hackers [4].

3.1 Examples of data slices for asset attribution

After training a model that translates from public data to a prediction of asset ownership, we assess the model. For the asset attribution problem, data slices might include:

Obvious outcomes. Models for attributing domains should be able to trivially associate flagship domains (`neurips.cc` belongs to the Neural Information Processing Systems Foundation). Mistakes on obvious data slices are a potential leading indicator of damaged user trust. We often use these "assertions" to block release of a model if accuracy on this slice is not 100%.

Experimental focuses. Data slices allow modelers to estimate the quantitative impact of underperformance that was reported anecdotally, like "this model seems to perform badly on assets that were recently sold to another company". Following modeler interventions on architecture, features, or train set sampling methods, data slices verify whether the interventions succeeded. This type of data slice is also longitudinally useful because it enables regression monitoring on challenging subsets.

Segmented data space. Data slices let us characterize models from a data-first perspective. We might separate assets with English-only public records from those with Spanish-only public records, create a data slice of "shadow IT" assets stood up by individuals independent of official processes, or a slice of assets belonging to "confuser" companies that share a name with a company of interest. These data slices facilitate quantitatively-based descriptions of model performance, and allow more nuanced understanding of how model alternatives perform that can inform which alternative should be deployed.

When releasing or comparing models, we compare data slice performance as well as precision, recall, and other metrics. We might characterize each model's performance on multiple dimensions with a single number each, and verify that performance on each dimension achieves a threshold. In other cases, we plot a histogram of $\hat{y}$ scores predicted by the model before and after some change, and verify that performance on the data slices moves in the expected direction.

3.2 Implementation in SimpleML

Our team uses SimpleML [14], an open-source BSD 3-Clause licensed ML management framework maintained by an author, to handle the complexity of a sprawling, rapidly iterated development environment. Data slices naturally fit into the SimpleML tracking paradigm, which provides a lineage tree and versions for each dataset. During evaluation, we use SimpleML to automatically select the latest version of each data slice. We can iterate quickly because of human-readable artifact names and fully defined lineage trees. We also use SimpleML's deployment patterns, which rely on tracked metadata to deploy exactly the intended artifact and its dependencies into production.

Using a framework like SimpleML is necessary because we have many simultaneous experiments that we evaluate against data slices, and each experiment produces streams of related dataset, transformation, model, and metric artifacts. Automatically tracking each version of an artifact, its lineage, and its package dependencies is critical to recovering artifact relationships in all non-trivial experiments. Reproducibility tracking cannot be managed by hand at scale. By automating the creation and evaluation of all models on known data slices, we reduce some aspects of the model and data management overhead. A robust management and evaluation framework also modularizes AI performance, so that teams can work independently while being protected from causing regressions in other teams' areas.

3.3 Use and limitations of data slices in a live system

Useful data slices illustrate regions of model underperformance. The transparency enabled by data slices can direct future development for lagging slices and surface biases against particular segments.

Data slices are a key part of trusting a continuous deployment workflow, in which new models automatically deploy into a production environment if all testing passes. Data slices can be used to gate release in two ways. Teams can require that performance on each data slice meets an objective threshold before deployment. This usage is a kind of assertion testing, automatically guarding releases from regressions or particularly egregious mistakes. Alternatively, performance on data slices can be aggregated according to a weighting function reflecting the overall objective. A reweighting function forms a single metric that enables comparing model versions along qualitatively-meaningful lines.

The primary limitation of using data slices in a production setting is knowing which data slices to monitor. In real-world scenarios there are often dozens or hundreds of potential slices. Models will rarely perform well across all data slices. It can be challenging to identify on which data slices to be willing to accept poor performance. To mitigate this, it is important to work with users and other stakeholders to understand which data slices best reflect practical modeling needs. Another challenge is that data slices may become irrelevant over time in cases of data drift [9]. It is crucial to do rigorous model monitoring in production and re-create data slices on a regular basis.

3.4 Ethical considerations

One can create data slices that emphasize the impact on specific groups of data, to understand how the model will affect those cohorts. For example, in cybersecurity we may want to have data slices that focus on content from different languages and geographic locations, so we can be sure that our algorithms perform well globally and are not overfitting to our local environment or optimizing for the largest cohort by sacrificing smaller cohorts.

Since the design of data slices is arbitrary, any unintended or intended biases of the person creating the data slices can potentially show up in the data slices. To avoid this, a diverse group should review the context of the data slices for unintended biases. It may also be useful to pair data slices. For instance, if the model should perform equally well at attributing assets to local community colleges as to top-tier universities, it may be useful to generate and monitor progress on data slices for both subsets, rather than focusing only on one data slice.

4 Conclusion

Data slices are a practical tool in multifaceted modern ML systems like those for internet asset attribution for computer network security. Data slices help us monitor and explain regression, iteratively improve models, and explain quantitative differences between models. Data slices enable us to systematically focus on improving data quality, understand coverage of classes, and build practical data evaluation systems that operate effectively beyond a single predictive test set.

By solving the "asset attribution" AI problem, we are able to identify dangerous unmanaged and accidentally misconfigured servers, support network compliance triage, and expose otherwise-hidden computer and network security issues. Our use of data slices improves our tuning of asset attribution models, which leads to remediation efforts by affected organizations and ultimately protects all the individuals, organizations, and governments that rely on the stability and security of the internet.

References

[1] S. Axelsson. The base-rate fallacy and the difficulty of intrusion detection. *ACM Trans. Inf. Syst. Secur.*, 3(3):186–205, Aug. 2000. ISSN 1094-9224. doi: 10.1145/357830.357849. URL `https://doi.org/10.1145/357830.357849`.

[2] A. L. Buczak and E. Guven. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys Tutorials*, 18(2):1153–1176, 2016. doi: 10.1109/COMST.2015.2494502.

[3] V. Chen, S. Wu, A. J. Ratner, J. Weng, and C. Ré. Slice-based learning: A programming model for residual learning in critical data slices. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper/2019/file/351869bde8b9d6ad1e3090bd173f600d-Paper.pdf`.

[4] D. Madory. The mystery of AS8003. 2021. URL `https://www.kentik.com/blog/the-mystery-of-as8003`.

[5] P. K. Manadhata and J. M. Wing. An attack surface metric. *IEEE Transactions on Software Engineering*, 37(3):371–386, 2010.

[6] M. Orcutt. Hijacking computers to mine cryptocurrency is all the rage. 2017. URL `https://www.technologyreview.com/2017/10/05/148783/hijacking-computers-to-mine-cryptocurrency-is-all-the-rage`.

[7] C. Osborne. Meris botnet assaults KrebsOnSecurity: The botnet appears to be made up of compromised routers. 2021. URL `https://www.zdnet.com/article/meris-botnet-assaults-krebsonsecurity`.

[8] N. Perlroth and S. Shane. In Baltimore and beyond, a stolen N.S.A. tool wreaks havoc. 2019. URL `https://www.nytimes.com/2019/05/25/us/nsa-hacking-tool-baltimore.html`.

[9] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence. *Dataset shift in machine learning*. The MIT Press, Cambridge, MA, 2009.

[10] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré. Snorkel. *Proceedings of the VLDB Endowment*, 11(3):269–282, Nov 2017. ISSN 2150-8097. doi: 10.14778/3157794.3157797. URL `http://dx.doi.org/10.14778/3157794.3157797`.

[11] I. A. Saeed, A. Selamat, and A. M. Abuagoub. A survey on malware and malware detection systems. *International Journal of Computer Applications*, 67(16), 2013.

[12] R. Sommer and V. Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *2010 IEEE Symposium on Security and Privacy*, pages 305–316, 2010. doi: 10.1109/SP.2010.25.

[13] C. Theisen, N. Munaiah, M. Al-Zyoud, J. C. Carver, A. Meneely, and L. Williams. Attack surface definitions: A systematic literature review. *Information and Software Technology*, 104: 94–103, 2018.

[14] E. Yadgaran. SimpleML. URL `https://github.com/eyadgaran/SimpleML`.