MoAT: Meta-Evaluation of Anti-Malware Trustworthiness

Sharon Lin
Max Planck Institute for Security and Privacy
Bochum, Germany
sharon.lin@mpi-sp.org

Christof Paar
Max Planck Institute for Security and Privacy
Bochum, Germany
christof.paar@mpi-sp.org

Abstract

Many studies have proposed methods for the automated detection of malware. The benchmarks used for evaluating these methods often vary, hindering a trustworthy comparative analysis of models. We analyzed the evaluation criteria of over 100 malware detection methods from 2018-2022 in order to understand the current state of malware detection. From our study, we devised several criteria for evaluating future malware detection methods. Our findings indicate that a finer-grained class balance in datasets is necessary to ensure the robustness of models. In addition, a metric robust to distribution shifts, e.g. PR-AUC, should be used in future studies to prevent the inflation of results in unrealistic distribution regimes. The composition of datasets should also be disclosed in order to ensure a fair comparison of models. To our knowledge, this study is the first to assess the trustworthiness of evaluations from multi-domain malware detection methods.

1 Introduction

In the last decade, the volume of malware distributed online has significantly increased. There are more than one billion malicious programs being distributed online, and more than half a million new programs detected every day [15]. Common types of malware, such as adware, spyware, and worms, increased by 200% in the past year. Newer attacks have also emerged, including malware that mines cryptocurrency on victims' devices or targets IoT devices [25].

Malware detection methods generally rely on file signatures and heuristics to identify malicious software. Repositories such as VirusTotal, Mapedia, and MalShare store file signatures from verified malware samples and match files based on existing signatures, while heuristic methods extract features from file contents and metadata and match based on rules that indicate whether a file is suspicious or not [3].

Though signatures can be a highly accurate signal of a malicious file, it is trivial for attackers to evade detection by inserting perturbations into their code. For heuristic analysis, the most common methods are static and dynamic analysis. Static analysis, which uses static data such as PE headers, compression ratio, or opcodes, is similarly easy to evade through binary obfuscation [26]. While dynamic analysis offers more direct insights into the execution of a program, it is also more time intensive and requires analysts to run files in an emulator or virtual environment to observe dynamic effects. Furthermore, dynamic analysis techniques can also be evaded through sandbox detection [2].

As the volume of distributed malware grows, comparing millions of files for similarities becomes an ever-increasing scalability challenge. More granular matching techniques have emerged from security research [10] in an effort to reduce the time required for analysts to classify malware. Thus, an opportunity has emerged for machine learning systems to assist in the defense against malicious cyberattacks. The dynamics of machine learning models are ideal for large-scale classification and recent work in the use of models for malware detection have demonstrated great promise.

Many papers on malware detection methods boast >95% accuracy in detection [1]. However, the benchmarks used for deriving this evaluation differ widely. While some authors used standardized platform-specific datasets to evaluate their malware detection methods, e.g. Drebin, others evaluated their methods on custom platform-agnostic datasets gathered from the wild, e.g. VirusTotal. These datasets are often closed-source, which further limits the reproducibility of their work. The metrics used to compare accuracy also vary across studies, from overall accuracy to Area Under the ROC Curve (AUC or AUC-ROC).

In other fields of applied machine learning, such as image recognition, benchmarks exist for comparing the results of studies, such as CartPole for reinforcement learning or ImageNet for image recognition. Malware detection lacks such a benchmark, due in part to the moving window effect of attackers routinely altering and introducing new forms of malware into the environment, posing a challenge for researchers to maintain a trustworthy standardized dataset. Nevertheless, as the volume of malware distribution continues to increase, large-scale models entrusted with protecting critical systems from cyberattacks will rely more on trustworthy evaluations, and the impact of maintaining a standard of evaluation across the field will become increasingly profound.

2 Related works

Several papers in the last few years have conducted comparative studies across various machine learning methods to detect different forms of malware. Challenges cited often include dataset class imbalance [3], a lack of open and public benchmarks [22], concept drift [14], and adversarial examples [7, 1].

A study by Ge et al. [11] analyzes the effects that imbalanced datasets can have on machine learning performance in malware detection. Their study specifically focused on Android malware and found that the performance of the models in other studies varied heavily depending on the datasets used for training and evaluation. In particular, by varying dataset composition by real-world and experimental settings, including quality, timeliness, and class imbalance between malware and benign files, they found that models tend to perform better on more balanced and recent datasets.

Liu et al. [23] also considered the lack of a common data source for Android malware detection. While datasets such as Drebin are among the largest and most cited [31], there are still numerous problems with data quality and timeliness, which can easily make malware detection models trained on these datasets obselete to newer attacks [4]. For instance, Drebin only contains samples from August 2010 to October 2012, MalGenome only contains samples from August 2010 to October 2011, and data from VirusShare and Contagio lack metadata such as descriptions, user ratings, and download numbers [18]. There is the further problem of sample duplication in the case where multiple datasets are used in order to obtain a larger corpus for training. Zhao et al. [33] investigated the effects of overlapping data between Genome, Drebin, AMD, and RmvDroid. Their study found that duplication rates between datasets ranged from <1% to 97.5%, and that performance achieved via training datasets with duplicate samples was always higher than that of datasets without duplicate samples.

Several more recent projects have also launched to address the dataset problem in malware detection. The Canadian Institute for Cybersecurity (CIC) released several datasets in the last few years, e.g. CICMalDroid 2020, in order to promote the use of fair and balanced training data for malware detection [9]. While our study doesn't introduce a new dataset, we investigated whether the evaluation criteria for state-of-the-art malware detection methods is trustworthy or if finer-grained criteria need to be established by the community. In the current literature, there is only one other study related to the trustworthiness of malware detectors [19]. In contrast to Kumar et al., we chose to evaluate not only hardware-based methods, but also multi-domain methods in order to conduct a comprehensive study across various platforms of malware to investigate the trustworthiness of malware detection model evaluation.

Table 1: Comparison of malware datasets

Dataset	Platform	# of studies	Date range	# of samples
VirusTotal	Various	22	2004 - Present	2,400,000,000
Drebin	Android	22	Aug 2010 - Oct 2012	5,560
VirusShare	Various	18	2012 - Present	51,646,084
AndroZoo	Android	10	2011 - 2016	20,998,632
MalGenome	Android	9	Aug 2010 - Oct 2011	1,260
CIC datasets	Various	9	2009 - 2022	1,900-400,000
AMD	Android	8	2010 - 2016	24,000
Contagio	Various	7	2008 - 2020	28,760
VXHeaven	Various	4	1995 - 2017	271,092
theZoo	Various	4	2014 - Present	351
Malicia Project	Windows	3	Mar 2012 - Feb 2013	11,688
Koodous	Android	3	2014 - Present	N/A
Ember	Windows	3	2016 - 2018	1,100,000
ClaMP	Windows	2	2011 - 2015	5,210
Sophos SOREL-20M	Windows	3	Jan 2017 - Apr 2019	20,000,000

3 Meta-evaluation of malware detectors

Our methodology considers the evaluation criteria for malware detection methods published from 2018-2022. Our dataset includes methodologies for detecting malware on Android, Windows, Linux, Mac OS, and various IoT operating systems.

The questions investigated in our study were:

- How trustworthy are the current state of evaluations of malware detection across all platforms and data types?
- How should models be evaluated in order to best measure the trustworthiness of their model's performance in real-world scenarios?

3.1 Datasets

We analyzed the evaluation criteria used across 150 studies proposing machine learning methods for malware detection. The papers were sourced from top-tier security conferences (USENIX, CCS, S&P), as well as search engines for scholarly literature (Google Scholar, Springer). Our search was limited to papers published in the last five years and only included those which proposed methods for malware detection using machine learning methods, excluding review and adversarial studies.

After surveying the studies, we extracted the highest-frequency datasets used by various malware detection methods and classified the datasets based on platform, data quality, timeliness (Table 1).

A significant number of studies used a custom-derived dataset. In most cases, this was because the study was using a non-standard data representation for malware detection, e.g. ciphertexts [30], memory images [24], steganography [29], microarchitectural side channels (bus cycles, cache misses, branch instructions) [20], or dynamic analysis traces (system logs, etc.) [6].

3.2 Metrics

Among the surveyed studies, the most common metric used for comparing model performance was accuracy, defined with true positive (TP), true negative (TN), false positive (FP), and false negative (FN) as

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

This was followed by the F1 score, which relates the harmonic mean of the precision and recall, which assists in mitigating biases caused by large outliers in either score:

$$
\begin{aligned}
F1 &= 2 * \frac{Precision * Recall}{Precision + Recall} \\
&= 2 * \frac{\frac{TP}{TP+FP} * \frac{TP}{TP+FN}}{\frac{TP}{TP+FP} + \frac{TP}{TP+FN}}
\end{aligned} \tag{2}
$$

The next most commonly used metrics were precision, recall, and AUC-ROC, which computes the area under the receiver operating characteristic curve in order to assess the performance of a binary classifier to discriminate across different thresholds.

In a survey study by Kouliaridis et al. [17], it was shown that due to differences in datasets and evaluation metrics, it was challenging to compare state-of-the-art methods in Android malware detection. Metrics from surveyed works included true positive rate (TPR), precision, recall, F1 score, accuracy, and AUC-ROC. This inconsistency resulted in inflated results, such as the case where accuracy is used on an imbalanced dataset where a high volume of true negatives contributes to higher scores. Of these metrics, AUC-ROC was argued to be the most robust measurement of effectiveness, as it remained conclusive even when datasets were imbalanced and was less sensitive to changes in class balance. This is because AUC-ROC scales with the total number of detected true values rather than the total number of correctly detected values [8]. Li et al. [21] proposed that the existence of undetected zero-days in benign datasets may pollute classification and render less robust metrics less meaningful in comparison.

Other metrics that are even more robust to class imbalance are PR-AUC and F1 score, which scale with the total number of correctly detected true values and are better representative of the performance requirements for anomaly detection to maximize true positives over minimizing false positives. If the size of the estimated positive sample set is much smaller than the size of the overall sample set, the differences across models compared with PR-AUC are much greater than with AUC-ROC. In particular, where AUC-ROC considers all samples in its scoring, PR-AUC disregards the performance of the classifier over true negatives.

4 Analysis

Risk Given the capabilities of machine learning systems to assist in large-scale cyberattacks and the relative lack of defense measures to counter such attacks, Hendrycks et al. [13] recommends research to be steered towards defense measures for machine learning systems. Our study follows in this direction, as we seek to improve the trustworthiness of large-scale machine learning models aimed at malware detection by critically evaluating the criteria used to measure their efficacy.

Even so, there are still risks associated with improving defense systems. The unification of datasets for accessing malware, even for the purpose of training more robust models, will ultimately allow attackers to access the same malware. Furthermore, open source methods will also make it easier for attackers to developer adversarial examples that evade detection from state-of-the-art methods. Nonetheless, data provenance is necessary for model interpretability, and evidence has shown that attackers will continue to exploit systems regardless of whether they are open-source or reverse-engineered.

Trustworthiness In the literature of machine learning, trustworthiness entails reliability on repeat decisions, fairness, robustness against input changes, explainability, and privacy preservation of sensitive data [32]. Our survey considers the trustworthiness of model evaluation in the context of malware detection methods. Data from the survey shows that the most commonly cited datasets are not necessary the most well-balanced across time and class distributions. Though well-calibrated models can resist differences in uncertainty over class distribution shifts, evidence shows that models trained on imbalanced datasets tend to have a detection bias towards the majority class[28]. Therefore, to ensure the reliability of models on independent and identically distributed (i.i.d.) data and robustness towards out-of-distribution (OOD) data, models should be trained on more balanced datasets.

The explainability of evaluations also varied, with many studies using metrics such as overall accuracy. Accuracy tends to obscure the ability of a model to detect malware in real-world scenarios, where the data may comprise a small percentage of the total online data. Furthermore, when various metrics are used for the evaluation of models, it can be more difficult to empirically compare their performance.

A common, interpretable benchmark that compares model performance in real-world scenarios addresses this challenge.

Regarding privacy, while evaluations of models do not tend to regard data privacy as a quantitative measure, privacy concerns do arise in the data acquisition of benign files, which is not always present in standardized malware datasets. Applications scraped from various marketplaces and search engines may contain sensitive or confidential data, so efforts to circumvent the use of private information should be taken to ensure the trustworthiness of models.

5 Conclusion

We analyzed the criteria used for evaluating multi-domain malware detection methods and provided a risk assessment for datasets and metrics used in evaluation. In addition, we devised guidelines for future practitioners to ensure the trustworthiness of models evaluation results. By unifying results found across platforms and domains, we hope to steer future malware detection studies towards greater trustworthiness.

Acknowledgments

We would like to thank our anonymous reviewers for their valuable feedback, which has helped to improve the quality of our work. We also gratefully acknowledge constructive input from Dr. Marc Fyrbiak and Dr. Erwin Quiring.

References

[1] F. A. Aboaoja, A. Zainal, F. A. Ghaleb, B. A. S. Al-rimy, T. A. E. Eisa, and A. A. H. Elnour. Malware detection issues, challenges, and future directions: A survey. *Applied Sciences*, 12 (17):8482, 2022. doi: https://doi.org/10.3390/app12178482.

[2] A. Afianian, S. Niksefat, B. Sadeghiyan, and D. Baptiste. Malware dynamic analysis evasion techniques: A survey. *ACM Computing Surveys*, 52(6):1–28, 2019. doi: http://dx.doi.org/10.1145/3365001.

[3] H. Aghakhani, F. Gritti, F. Mecca, M. Lindorfer, S. Ortolani, D. Balzarotti, G. Vigna, and C. Kruegel. When malware is packin' heat; limits of machine learning classifiers based on static analysis features. *Network and Distributed Systems Security (NDSS) Symposium*, 2020. doi: https://dx.doi.org/10.14722/ndss.2020.24310.

[4] A. Batouche and H. Jahankhani. A comprehensive approach to android malware detection using machine learning. *Journal of Information Security and Applications*, 2021. doi: https://doi.org/10.1007/978-3-030-72120-6_7.

[5] G. W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 1950. doi: https://doi.org/10.1175/1520-0493(1950)078<0001:VOFEIT>2.0.CO;2.

[6] P. Calderon, H. Hasegawa, Y. Yamaguchi, and H. Shimada. Malware detection based on https characteristic via machine learning. *International Conference on Information Systems Security and Privacy (ICISSP)*, 2018. doi: https://doi.org/10.5220/0006654604100417.

[7] L. Chen, Y. Ye, and T. Bourlai. Adversarial machine learning in malware detection: Arms race between evasion attack and defense. *European Intelligence and Security Informatics Conference (EISIC)*, 2017. doi: https://doi.org/10.1109/EISIC.2017.21.

[8] T. Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006. doi: https://doi.org/10.1016/j.patrec.2005.10.010.

[9] C. I. for Cybersecurity. Datasets. URL `https://www.unb.ca/cic/datasets/index.html`.

[10] D. French. Fuzzy hashing against different types of malware. URL `https://insights.sei.cmu.edu/blog/fuzzy-hashing-against-different-types-of-malware/`.

[11] X. Ge, Y. Huang, Z. Hui, X. Wang, and X. Cao. Impact of datasets on machine learning based methods in android malware detection: an empirical study. *IEEE 21st International Conference on Software Quality, Reliability and Security (QRS)*, 2021. doi: https://doi.org/10.1109/QRS54544.2021.00019.

[12] R. Harang and E. M. Rudd. Sorel-20m: A large scale benchmark dataset for malicious pe detection. *Proceedings of the Conference on Applied Machine Learning for Information Security*, 2021.

[13] D. Hendrycks, N. Carlini, J. Schulman, and J. Steinhardt. Unsolved problems in ml safety. *arXiv*, 2022. doi: https://doi.org/10.48550/arXiv.2109.13916.

[14] D. Hu, Z. Ma, X. Zhang, P. Li, D. Ye, and B. Ling. The concept drift problem in android malware detection and its solution. *Security and Communication Networks*, 2017. doi: https://doi.org/10.1155/2017/4956386.

[15] B. Jovanovic. A not-so-common cold: Malware statistics in 2022. URL `https://dataprot.net/statistics/malware-statistics/`.

[16] A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Conference on Neural Information Processing Systems (NIPS)*, 2017. doi: https://dl.acm.org/doi/10.5555/3295222.3295309.

[17] V. Kouliaridis and G. Kambourakis. A comprehensive survey on machine learning techniques for android malware detection. *Information*, 2021. doi: https://doi.org/10.3390/info12050185.

[18] M. Krzysztoń, B. Bok, M. Lew, and A. Sikora. Lightweight on-device detection of android malware based on the koodous platform and machine learning. *Sensors*, 22(17):6562, 2022. doi: https://doi.org/10.3390/s22176562.

[19] H. Kumar, N. Chawla, and S. Mukhopadhyay. Towards improving the trustworthiness of hardware based malware detector using online uncertainty estimation. *ACM/IEEE Design Automation Conference (DAC)*, 2021. doi: https://doi.org/10.1109/DAC18074.2021.9586288.

[20] A. Kwan. Malware detection at the microarchitecture level using machine learning techniques. Master's thesis, California State University, Long Beach, 2020.

[21] Y. Li, D. Caragea, L. Hall, , and X. Ou. Experimental study of machine learning based malware detection systems' practical utility. *HICSS Symposium on Cybersecurity Big Data Analytics*, 2020.

[22] G. Liang, J. Pang, Z. Shan, R. Yang, and Y. Chen. Automatic benchmark generation framework for malware detection. *Security and Communication Networks*, 2018. doi: https://doi.org/10.1155/2018/4947695.

[23] K. Liu, S. Xu, G. Xu, M. Zhang, D. Sun, and H. Liu. A review of android malware detection approaches based on machine learning. *IEEE Access*, 2020. doi: https://doi.org/10.1109/ACCESS.2020.3006143.

[24] S. Lyles, M. Desantis, J. Donaldson, M. Gallegos, H. Nyholm, C. Taylor, and K. Monteith. Machine learning analysis of memory images for process characterization and malware detection. *52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, 2022. doi: https://doi.org/10.1109/DSN-W54100.2022.00035.

[25] Malwarebytes. Malwarebytes 2022 threat review. URL `https://www.malwarebytes.com/resources/malwarebytes-threat-review-2022/index.html`.

[26] A. Moser, C. Kruegel, and E. Kirda. Limits of static analysis for malware detection. *Twenty-Third Annual Computer Security Applications Conference (ACSAC)*, 2007. doi: https://doi.org/10.1109/ACSAC.2007.21.

[27] M. P. Naeini, G. F. Cooper, and M. Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. *AAAI Conference on Artificial Intelligence*, 2015. doi: https://dl.acm.org/doi/10.5555/2888116.2888120.

[28] Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. V. Dillon, B. Lakshminarayanan, and J. Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Conference on Neural Information Processing Systems (NIPS)*, 2019. doi: https://dl.acm.org/doi/10.5555/3454287.3455541.

[29] H. D. Samuel, M. S. Kumar, R. Aishwarya, and G. Mathivanan. Automation detection of malware and stenographical content using machine learning. *6th International Conference on Computing Methodologies and Communication (ICCMC)*, 2022. doi: https://doi.org/10.1109/ICCMC53470.2022.9754063.

[30] J. Son, E. Ko, U. B. Boyanapalli, D. Kim, Y. Kim, and M. Kang. Fast and accurate machine learning-based malware detection via rc4 ciphertext analysis. *International Conference on Computing, Networking and Communications (ICNC)*, 2019. doi: https://doi.org/10.1109/ICCNC.2019.8685644.

[31] V. Syrris and D. Geneiatakis. On machine learning effectiveness for malware detection in android os using static analysis data. *Journal of Information Security and Applications*, 59, 2021. doi: https://doi.org/10.1016/j.jisa.2021.102794.

[32] E. Toreini, M. Aitken, K. Coopamootoo, K. Elliott, C. G. Zelaya, and A. van Moorsel. The relationship between trust in ai and trustworthy machine learning technologies. *Conference on Fairness, Accountability, and Transparency*, 2020. doi: https://doi.org/10.1145/3351095.3372834.

[33] Y. Zhao, L. Li, H. Wang, H. Cai, T. F. Bissyandé, J. Klein, and J. Grundy. On the impact of sample duplication in machine-learning-based android malware detection. *ACM Transactions on Software Engineering and Methodology*, 30(3):1–38, 2021. doi: https://doi.org/10.1145/3446905.

A Appendix

A.1 Evaluating trustworthiness

Within the machine learning literature, trustworthiness of a model is understood to be measured by the quality of the model's predictive uncertainty. When measuring the uncertainty of a classification model, the following equation may be used [16]:

$$P\left(y \mid \mathrm{x}^{*}, \mathcal{D}\right)=\int P\left(y \mid \mathrm{x}^{*}, \Theta\right) P(\Theta, \mathcal{D}) d \Theta \tag{1}$$

where y is the classification label, $x*$ is the test data, D is the training dataset, and Θ is the model parameters for the distribution over which a Bayesian model learns (the posterior).

Our investigation was concerned with the impact of different dataset compositions on the trustworthiness of model evaluation. We were less concerned with the impact of dataset distribution shifts on the empirical performance of individual models, although a study by Ovadia et al. [28] found that for all models, regardless of method, performance was consistently worse with increasing distribution shift (though deep ensembles were the least affected).

A.2 Robustness through model calibration

In practice, especially for online models which observe unpredictable shifts in data distribution, robustness under out-of-distribution (OOD) inputs is important for safe deployment. Calibration on a hold-out set is a common practice to enable accurate assessment of risk, as it allows practitioners to evaluate how accuracy degrades over time and allows a model to abstain from making decisions with low confidence.

Probability scoring can be computed with the Brier Score [5]:

$$BS=|Y|^{-1} \sum_{y \epsilon Y}\left(p(y|x_n, \theta)-\delta(y-y_n)\right)^2 \tag{3}$$

where $p(y|x_n, \theta)$ is the predicted probability vector and y_n is the one-hot encoded true response, or with the Expected Calibration Error (ECE) [27]:

$$ECE=\sum_{s=1}^{S} \frac{|B_s|}{n}|acc(B_s)-conf(B_s)| \tag{4}$$

which measures the correspondence between predicted probabilities and empirical probabilities.

Findings by Ovadia et al. showed that calibration on an independent and identically distributed (i.i.d.) dataset, e.g. on a test set, did not result in better calibration under dataset shift.

A.3 Comparison of malware datasets

In our evaluation, we compared characteristics of commonly used datasets for malware detection. To expand upon the differences between these datasets, several distinguishing factors are discussed.

Platform There are numerous platforms for which malware can be targeted. Some of the datasets in our study, including Drebin, AndroZoo, and AMD, specifically contain files formatted to one class of platforms, namely Android operating systems. Others, such as VirusTotal, contain a larger selection of file formats, including Microsoft Advanced Streaming/Systems Format (ASF) files, Java .class bytecode files, compressed files, ISO image files, and Apple MachO files.

Date range Malware is most often deployed for currently supported platforms. Although system architectural vulnerabilities may persist through updates and users may continue to use older versions of platforms, training on older malware datasets poses the risk of data drift. Some datasets that not actively maintained, such as the Malicia Project, are explicitly deprecated for this reason.

Metadata Along with file contents, file metadata is often used in static analysis. The method of data aggregation for datasets affects the availability of this metadata. For instance, Android files on VirusTotal may contain API data, but are missing app descriptions, user ratings, and install statistics that are available for files downloaded from Google Play.

Dataset size Obtaining a large corpus of balanced data is important for training accurate detection models. Commercial malware detection models can be trained on tens to hundreds of millions of samples [12]. From the datasets we surveyed, the largest corpus was VirusTotal with 2.4 billion samples, and the smallest was theZoo with 351 samples. A full table of dataset sizes is available here.

Reliability For crowd-sourced datasets, such as Koodous, tag reliability is a potential risk. The platform ranks files based on user-generated descriptions and votes, with no safety measures for data pollution from malicious actors.

A.4 Takeaways and recommendations

From our study, the following best practices are recommended for future practitioners to ensure the trustworthiness of evaluation results:

- Disclose the origins of datasets or to make available custom derived datasets whenever possible, i.e. when privacy measures are taken to protect sensitive data used for training.
- Disclose the constraints of the data being detected by a model (e.g. Windows file binaries or Android APKs) rather than broadly classifying generally malicious files. Even if the detection method is aimed at detecting malicious files agnostic of platform, composition should still be detailed in the methodology to aid in the evaluation of class balance.
- Disclose how benign files were obtained, especially in the case of custom datasets extracted from the wild, in order to reduce the risk of accessing sensitive or confidential data.
- Balance datasets used for training and testing over time and classes (e.g. benign/malicious or adware/banking/SMS malware/riskware/benign for a more fine-grained classification) to minimize the effects of distribution shift.
- Compare the performance of models using ranking metrics such as F1 score or PR-AUC, especially if datasets are imbalanced.

A.5 Artifacts

The dataset of papers surveyed in this study is available at this link.