







Towards Explainable Malware Detection with Structured Machine Learning

Extended Abstract

Peter Švec¹ , Tomáš Bisták² , Martin Homola² , Štefan Balogh¹ ,
Ján Klůka² , and Alexander Šimko² 

¹ Institute of Computer Science and Mathematics, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology, Ilkovičova 3, 812 19 Bratislava, Slovakia

{peter.svec1, stefan.balogh}@stuba.sk

² Department of Applied Informatics, Faculty of Mathematics, Physics and Informatics, Comenius University, Mlynská dolina, 842 48 Bratislava, Slovakia

{kluka, homola, alexander.simko}@fmph.uniba.sk, bistak5@uniba.sk

Abstract. We describe a use case in the domain of malware detection, complemented by a semantically tailored version of the well-known EMBER dataset, including fractional datasets of different sizes. We then report on our first results applying structured machine learning in form of DL concept learning on this data.

Keywords: Explainability · Malware detection · Concept learning

1 Introduction

Structured Machine Learning (SML) was described as *symbolic supervised machine learning from structured data* [25]. Given a formal language \mathcal{L} , in which expressions ϕ (e.g. conditions, rules, concept expressions, etc.) can be evaluated as true or not true w.r.t. any given data example e ($e \models \phi$ or $e \not\models \phi$, respectively), the task is to learn expressions ψ s.t. $e \models \psi$ for all positive examples $e \in E^+$ and $e \not\models \psi$ for all negative examples $e \in E^-$. Such learned expressions can then be used as classifiers (on previously unseen examples) but depending on their quality, complexity, and size, they may also serve as interpretable explanations or justifications for the classified part of the sample. Rooting such expressions in a commonly agreed vocabulary of a suitable domain ontology may further improve their interpretability. It may also be valuable for some applications that many of these methods allow to find such characteristic expressions independently and thus may be applied also on top of classifications obtained by black-box ML, by heuristic analysis, or by human expertise.

There are many potential application domains for SML; we were able to find about 20 datasets explored in 4 published works [16,17,25,15]. The largest dataset had 17,941 samples [16], the second largest had 2,567 samples [16]. Only one work attempted to provide a standardized set of use cases, easy to reuse [25]. We argue that introducing more use cases from real-world domains with much larger datasets will help to:

1. Create a challenge for more effective algorithms and tools.
2. Create a challenge for

their users to deal with noisy data, large sample sizes, and other problems posed by real-world datasets.

We propose the area of malware detection as a novel use case for SML methods. We believe that this is beneficial for malware researchers – applications of machine learning in this area are spiking [23,5,18] and the need to improve their interpretability is now a recognized issue [13,7,12,3]. Jointly, to SML researchers this will introduce a new and unique use case based on large real-world datasets that may contribute to evaluation and improvement of their algorithms and tools.

We accompany the use case with benchmarking data in the form of an *ontology* and *annotated datasets* that have been extracted and adopted from the popular EMBER dataset [1]. We also briefly outline our first results by applying concept learning (a form of SML) on the data.

2 Use Case

Application domain: We propose the domain of malware detection, in which, as we documented, concept learning and similar methods are inherently relevant to address real malware research problems.

Data sources: Datasets such as EMBER [1] and SOREL [6] provide sufficiently large samples of data (around 800k and 15M annotated samples). These datasets are real-world, sufficiently rich, well structured, and well-known to malware researchers.

Ontology: We have provided the PE Malware ontology [20], a suitable reference ontology rooted in expert knowledge over which descriptive characterizations can be constructed.

Datasets: Jointly we released semantic data sets obtained by translating the EMBER data to the RDF format, easy to feed into any concept learning tool. They are available in different sizes and can be used and referred to by different experiments to allow for comparisons. A number of works resorted to reducing, e.g., EMBER in some (non-canonical) way which makes comparisons difficult [24,11,4].

Tasks: The task to address on this datasets is to generate descriptive expressions (e.g., DL concepts) that characterize the malware (benign) instances annotated as the positive (negative) examples in the sample. The learned expressions should be evaluated based on (a) the time required to reach them, (b) the achieved precision that can be compared with results obtained from state-of-the-art ML classifiers; and nonetheless (c) their interpretability and meaningfulness to malware experts.

3 Datasets and Ontology

3.1 EMBER and SOREL datasets

EMBER [1] and SOREL [6] are datasets for training static malware detection models. They contain structured data entries of Windows Portable Executable (PE) files. EMBER and SOREL contain around 800k and 20M entries labeled as malware or benign, respectively. Each entry has properties such as: file size; number of imported and exported functions; target architecture; information about PE file’s sections such as section

name, content type, access rights; list of imported functions per DLL; list of exported functions; various histograms and statistics, and more. EMBER data are available in the form of JSON objects, SOREL data in the form of SQLite3 and LDBM database entries.

3.2 PE Malware Ontology

To semantically capture EMBER and similar sources with static analysis data on malware and benign PE files, we have designed the *PE Malware Ontology*³. It is a lightweight OWL 2 ontology expressed within $DL\text{-}Lite_{core}(D)$, i.e., the OWL 2 QL profile. While it evolved from the structure of EMBER, it is not a direct abstraction of the schema of EMBER or any other particular dataset. In total, the ontology comprises 195 classes, 6 object properties, and 10 data properties. We are giving a brief overview of the ontology here and refer the reader interested in more details to our technical report [20].

Focusing on interpretability, the ontology represents PE file structure and mostly qualitative features that make sense from a malware expert’s point of view. Many quantitative features (file sizes, byte histograms, etc.) are disregarded. Although some were found to be statistically useful for ML-based malware detection [14], they are difficult to interpret and can lead to classifiers susceptible to trivial adversarial attacks [19]. However, we have added two qualitative features, threshold-based on quantitative features known to be significant and easily interpretable to malware experts (a low number of imports and a high section content entropy).

To facilitate reproduction of experimental results, we have created a suite of RDF data sets based on the PE Malware Ontology and EMBER and SOREL data.

The dataset `dataset_1_800000.owl` describes all labeled EMBER samples. Since SML is computationally intensive, we provide fractional datasets of 1 k, 10 k, and 100 k randomly selected samples, keeping EMBER’s 50:50 malware-to-benign ratio. Ten variants of each size (`dataset_N_size.owl` for $N \in \{1, \dots, 10\}$) have been produced to help researchers compensate for the selection bias. In order to encourage the k -fold evaluation methodology we did not provide splits into the training and testing set.

As for the SOREL dataset, we created four fractional datasets, each consisting of 1 k of samples drawn from the entire SOREL at random. Each of these datasets preserves the above established 50:50 ratio of malware- to benign-software examples, but also ensures an equal share of EXEs and DLLs within both the malicious and the benign samples. We touch upon why we introduced the latter restriction later in Section 4.2.

4 Results

4.1 EMBER Data

We experimented with four learning algorithms: OCEL [9], CELOE [10], PARCEL [21] and SPARCEL [22], all implemented in *DL-Learner* [8], a state-of-the-art framework for supervised machine learning in description logics [2]. Concept learning is computationally demanding, thus in order to establish a baseline of results in this domain, we

³ <https://github.com/orbis-security/pe-malware-ontology>

started with the 1 k datasets from our suite. For hyper-parameter calibration, we randomly selected dataset `dataset_8_1000.owl` and for validation, we selected datasets 1–5 of the same size. We used the k -fold cross-validation technique for evaluation, with $k = 5$.

We calibrated four hyper-parameters: noise, the use of `hasValue` and negation constructors in the generated descriptions, and the cardinality limit. The best results during the calibration phase were achieved using PARCEL (with noise 1, `hasValue` enabled, negation disabled and cardinality limit set to 10) and SPARCEL (with noise 1, `hasValue` enabled, negation disabled and cardinality limit set to 20), achieving F1 scores of 0.76 and 0.77, respectively.

The results from validation were quite similar, however, accuracy and F1 were lower in most cases. Both parallel algorithms, PARCEL and SPARCEL, achieved an F1 score of 0.72, while the nonparallel learners, OCEL and CELOE, achieved an F1 score of 0.70.

Expression (1) was produced by the PARCEL algorithm and can be interpreted as a PE file with at least two sections, both with a high entropy and a nonstandard name. Class expression (2) was acquired from OCEL and most likely indicates a packed executable. From the explainability point of view, such expressions are plausible descriptions of possible malware samples.

$$\begin{aligned} &\geq 2 \text{ has_section.} (\exists \text{ has_section_feature.} \{ \text{high_entropy} \} \\ &\quad \sqcap \exists \text{ has_section_feature.} \{ \text{nonstandard_section_name} \}) \end{aligned} \quad (1)$$

$$\begin{aligned} &\text{ExecutableFile} \\ &\quad \sqcap \exists \text{ has_section.} \exists \text{ has_section_feature.} \text{HighEntropy} \sqcap \text{WriteExecuteSection} \end{aligned} \quad (2)$$

4.2 SOREL Data

A follow-up study aimed to validate the suitability of our approach on the SOREL dataset. After some initial tests on a random sample of 500 malicious and 500 benign examples, we noticed that the nonparallel learners were covering predominantly EXE malware with their descriptions due to the scarcity of (malicious) DLLs in SOREL, and thus in our sample as well. Therefore, we prepared the four balanced datasets described in Section 3. We also detected several inconsistencies and flaws in the behavior of the learning algorithms during preliminary experimentation, which forced us to correct and improve their implementation prior to beginning the study. These modifications, among other factors, motivated us to change the set of hyper-parameters for optimization, more specifically, we calibrated noise, the use of negations and the “some-only” rule⁴, and either limited the cardinality constraints exclusively to the `has_section` property or forbade at-most restrictions with a cardinality higher than 1. The calibration was performed on one of the four prepared datasets and a subsequent validation on the remaining three.

The obtained calibration results confirmed the superiority of the parallel algorithms over the nonparallel ones observed in the study on EMBER data. The best PARCEL configuration (with noise set to 0, negations and the “some-only” rule disabled, and no restrictions on cardinality constraints) achieved an F1 score of 0.77, while the best

⁴ For more details, refer to the documentation of DL-Learner.

SPARCEL setup (with noise set to 0, negations and the “some-only” rule enabled, and no restrictions on cardinality constraints) reached an F1 score of 0.76.

Unlike in the validation phase for the first study, the majority of the examined algorithms showed relatively stable behavior during the validation here, equaling their performance from the preceding calibration in terms of evaluation metrics. We largely attribute this to the implementation changes we made.

Looking at the final descriptions, we can conclude that balancing the datasets w.r.t. the number of EXEs and DLLs among the malicious and the samples fulfilled its purpose. Both OCEL and CELOE endeavored to find disjunctions with one disjunct for EXEs and one (not just) for DLLs, e.g., the disjunction (3) discovered by OCEL. However, the need for a top-level disjunction also prevented OCEL and CELOE from getting to more elaborate descriptions of either EXE or DLL malware such as (2), since it essentially caused them to search in two directions simultaneously. The parallel algorithms were not affected by this altered composition of datasets and approached the problem of malware characterization as in the first study.

$$\begin{aligned} & (\text{ExecutableFile} \sqcap \exists \text{has_file_feature. MultipleExecutableSections}) \\ & \sqcup (\exists \text{has_action. } \neg \text{SendHttpRequest} \\ & \quad \sqcap \forall \text{has_file_feature. } (\neg \text{Signature} \sqcap \neg \text{Symbols})) \end{aligned} \quad (3)$$

Overall, we may say that the results of the second study proved that concept learning can be deemed a valid method of malware characterization and that the investigated algorithms represent a decent baseline.

5 Conclusions

Exploitation of SML methods may bring significant impact on explainability in malware detection. Therefore we have proposed a use case for SML in this domain, accompanied by a dataset derived from EMBER [1]. The dataset contains real-world data and offers unique challenges for SML also due to its size.

Preliminary application of concept learning shows interesting results. The learned concepts obtained so far in our experiments did not discover any novel patterns from the viewpoint of malware analysis, but they show to be well understandable by domain experts. The achieved accuracy measures are not yet comparable to state-of-the-art machine learning classifiers (which are not explainable). This is partly due to not being able to process larger fractions of the data, partly due to intentionally excluding features inherently not explainable (such as byte histograms), and also for not being able to process numeric values well by crisp concept learning. There is a large number of interesting issues for future research: improvement of the data representation, handling numeric values by fuzzy concept learning or other SML method, handling larger volumes by improving the concept learning algorithms but also by the division of the dataset based on the inherent structure of the data (e.g. by different malware families).

Acknowledgements. This research was sponsored by the Slovak Republic under the grant APVV-19-0220 (ORBIS) and by the EU under the H2020 grant no. 952215 (TAILOR) and under Horizon Europe grant no. 101079338 (TERAIS).

References

1. Anderson, H.S., Roth, P.: EMBER: An open dataset for training static PE malware machine learning models. arXiv preprint arXiv:1804.04637 (2018)
2. Bühmann, L., Lehmann, J., Westphal, P., Bin, S.: DL-learner structured machine learning on semantic web data. In: Companion Proceedings of the The Web Conference 2018. pp. 467–471 (2018)
3. Dolejš, J., Jureček, M.: Interpretability of machine learning-based results of malware detection using a set of rules. In: Stamp, M., Aaron Visaggio, C., Mercaldo, F., Di Troia, F. (eds.) Artificial Intelligence for Cybersecurity. pp. 107–136. Springer International Publishing, Cham (2022). https://doi.org/10.1007/978-3-030-97087-1_5
4. Ghouti, L., Imam, M.: Malware classification using compact image features and multiclass support vector machines. IET Inf. Secur. **14**(4), 419–429 (2020). <https://doi.org/10.1049/iet-ifs.2019.0189>
5. Gibert, D., Mateu, C., Planes, J.: The rise of machine learning for detection and classification of malware: Research developments, trends and challenges. Journal of Network and Computer Applications **153**, 102526 (2020)
6. Harang, R., Rudd, E.M.: SOREL-20M: A large scale benchmark dataset for malicious PE detection. arXiv preprint arXiv:2012.07634 (2020)
7. Iadarola, G., Martinelli, F., Mercaldo, F., Santone, A.: Towards an interpretable deep learning model for mobile malware detection and family identification. Computers & Security **105**, 102198 (2021). <https://doi.org/10.1016/j.cose.2021.102198>
8. Lehmann, J.: DL-learner: Learning concepts in description logics. The Journal of Machine Learning Research **10**, 2639–2642 (2009). <https://doi.org/10.5555/1577069.1755874>
9. Lehmann, J.: Learning OWL Class Expressions, Studies on the Semantic Web, vol. 6. IOS Press (2010). <https://doi.org/10.3233/978-1-61499-340-7-i>
10. Lehmann, J., Auer, S., Bühmann, L., Tramp, S.: Class expression learning for ontology engineering. J. Web Semant. **9**(1), 71–81 (2011). <https://doi.org/10.1016/j.websem.2011.01.001>
11. Liu, X., Lin, Y., Li, H., Zhang, J.: A novel method for malware detection on ML-based visualization technique. Comput. Secur. **89** (2020). <https://doi.org/10.1016/j.cose.2019.101682>
12. Marais, B., Quertier, T., Chesneau, C.: Malware analysis with artificial intelligence and a particular attention on results interpretability. In: Matsui, K., Omatu, S., Yigitcilar, T., Rodríguez-González, S. (eds.) Distributed Computing and Artificial Intelligence, Volume 1: 18th International Conference, DCAI 2021, Salamanca, Spain, 6-8 October 2021. LNNS, vol. 327, pp. 43–55. Springer (2021). https://doi.org/10.1007/978-3-030-86261-9_5
13. Mills, A., Spyridopoulos, T., Legg, P.: Efficient and interpretable real-time malware detection using random-forest. In: 2019 International conference on cyber situational awareness, data analytics and assessment (Cyber SA). pp. 1–8. IEEE (2019)
14. Oyama, Y., Miyashita, T., Kokubo, H.: Identifying useful features for malware detection in the ember dataset. In: 2019 Seventh International Symposium on Computing and Networking Workshops (CANDARW). pp. 360–366 (2019). <https://doi.org/10.1109/CANDARW.2019.00069>
15. Rizzo, G., Fanizzi, N., d’Amato, C.: Class expression induction as concept space exploration: From DL-FOIL to DL-FOCL. Future Generation Computer Systems **108**, 256–272 (2020)
16. Rizzo, G., Fanizzi, N., d’Amato, C., Esposito, F.: A framework for tackling myopia in concept learning on the web of data. In: Knowledge Engineering and Knowledge Management:

- 21st International Conference, EKAW 2018, Nancy, France, November 12-16, 2018, Proceedings 21. pp. 338–354. Springer (2018)
17. Sarker, M.K., Hitzler, P.: Efficient concept induction for description logics. In: The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019. pp. 3036–3043. AAAI Press (2019)
 18. Shaukat, K., Luo, S., Varadharajan, V., Hameed, I.A., Xu, M.: A survey on machine learning techniques for cyber security in the last decade. *IEEE Access* **8**, 222310–222354 (2020)
 19. Suci, O., Coull, S.E., Johns, J.: Exploring adversarial examples in malware detection. In: 2019 IEEE Security and Privacy Workshops (SPW). pp. 8–14. IEEE (2019)
 20. Švec, P., Balogh, Š., Homola, M., Klůčka, J.: Knowledge-based dataset for training PE malware detection models. arXiv preprint arXiv:2301.00153 (2022)
 21. Tran, A.C., Dietrich, J., Guesgen, H.W., Marsland, S.: An approach to parallel class expression learning. In: International Workshop on Rules and Rule Markup Languages for the Semantic Web. pp. 302–316. Springer (2012)
 22. Tran, A.C., Dietrich, J., Guesgen, H.W., Marsland, S.: Parallel symmetric class expression learning. *The Journal of Machine Learning Research* **18**(1), 2145–2178 (2017)
 23. Ucci, D., Aniello, L., Baldoni, R.: Survey of machine learning techniques for malware analysis. *Computers & Security* **81**, 123–147 (2019). <https://doi.org/https://doi.org/10.1016/j.cose.2018.11.001>, <https://www.sciencedirect.com/science/article/pii/S0167404818303808>
 24. Vinayakumar, R., Alazab, M., Soman, K.P., Poornachandran, P., Venkatraman, S.: Robust intelligent malware detection using deep learning. *IEEE Access* **7**, 46717–46738 (2019). <https://doi.org/10.1109/ACCESS.2019.2906934>
 25. Westphal, P., Bühmann, L., Bin, S., Jabeen, H., Lehmann, J.: SML-Bench—A benchmarking framework for structured machine learning. *Semantic Web* **10**(2), 231–245 (2019)