

On the Prospects of \mathcal{EL} and \mathcal{ELU} Concept Learners for Explainable Malware Detection

Martin Demovič¹ , Peter Švec² , Martin Homola¹ , and Maurice Funk³ 

¹ Comenius University in Bratislava, Mlynská dolina, 84248 Bratislava, Slovakia
demovic17@uniba.sk, homola@fmph.uniba.sk

² Slovak Technical University, Ilkovičova 3, 84104 Bratislava, Slovakia
peter.svec1@stuba.sk

³ Leipzig University, Augustusplatz 10, Leipzig, 04109, Germany

Abstract. We compare concept learners DL-Learner, SPELL and ALC-SAT on real-world data based on the EMBER malware dataset with the focus on learning \mathcal{EL} and \mathcal{ELU} concepts. We observe that the latter two systems are promising computationally; however, the DL-Learner’s ELTL algorithm achieved the highest F1 measure, albeit at the expense of increased false positive rate.

Keywords: Explainable AI · Malware analysis · Concept learning.

1 Introduction

Structured machine learning (SML) [31] was applied in various domains to learn *symbolic classifiers* [5, 6, 31] – logical formulas that are true if a given input sample is classified under the learned category and false otherwise. Compared to black-box machine learning (ML) [15], such a classifier is *inherently* interpretable by the user, who may thus understand the reasons why the sample was classified as true or false; offering better, intrinsic explainability [26], than e.g. popular post-hoc methods such as LIME [24] and SHAP [20].

The overwhelming amounts of malware data inevitably lead to the employment of ML-based malware classifiers [12, 27, 28]. Given high mission-criticality, the community soon recognized that black-box methodology is of limited use [8, 21, 22]. SML is one of the promising approaches to overcome the issue, and given the availability of ontologies that provide natural shared vocabularies to describe malware data [32], *concept learning* [3, 4, 9, 19, 25] is a natural choice. On the other hand, it is computationally expensive. Black-box ML methods are able to handle datasets of millions of samples [1, 14]. Previous experiments with concept-learning showed that a state-of-the-art system, DL-Learner [3], already struggled to process datasets of 10–20 thousand samples.

Expressive concepts supported by DL-Learner (up to *SHOIQ* [16]) may be less comprehensible to human users (particularly those involving value (\forall) and less-than (\leq) number restrictions) [30, 32]. Here, we focus on the lightweight

\mathcal{EL} [2], allowing conjunction (\sqcap) and existential restriction (\exists), enabling naturally comprehensible concepts that capture relevant patterns in data. More importantly, \mathcal{EL} exhibits lower (polynomial-time) complexity of reasoning [2, 13], which the concept learner needs to execute for every candidate concept. While adding disjunction (\sqcup) – and thus reaching \mathcal{ELU} – causes the complexity of reasoning to jump to EXPTIME , previous research [30, 32] indicates that it is useful, as it enables expressing multiple options in matching. Notably, a smaller set of constructors – compared to logics such as \mathcal{SHOIQ} – also decreases the search space that the concept learner has to explore. While learning in \mathcal{EL} and \mathcal{ELU} results in simpler concepts, they may be found more efficiently, and thus it would be possible to process larger datasets.

2 Compared systems

DL-Learner offers multiple concept learning algorithms. Expressive (\mathcal{SHOIQ}) algorithms were applied in malware detection [32]: OCEL prioritizing the evaluation metrics, and CELOE prioritizing shorter concepts (useful in ontology engineering) [3]. However, DL-Learner also features dedicated algorithms ELTL and DELTL, for \mathcal{EL} and \mathcal{ELU} [17, 18].⁴ All rely on heuristic search, starting from the most general concept \top and consecutively generating more specific concepts by a so-called refinement operator. The search is infinite, but the approach guarantees that no concepts are missed on the way.

SPELL [4] is a dedicated learner for \mathcal{EL} concepts based on *bounded fitting*: iteratively increasing the size bound of the searched concept until a fitting is found. Drawing from probably approximately correct (PAC) learning [29], it formally guarantees generalization from the training data to previously unseen data, and in the spirit of Occam algorithms, also hypotheses of small size. The system relies on a translation to SAT solvers, leveraging their practical efficiency.

More recently, the same approach was extended up to \mathcal{ALC} in the system called ALC-SAT [11]. For comparison with DELTL, we configured ALC-SAT to learn \mathcal{ELU} concepts in our experiments.

3 Datasets and Methodology

Previous experiments [4, 11] with SPELL and ALC-SAT were conducted on datasets with up to 1000 examples. Real-world malware datasets are often significantly larger. For our experiments, we draw on the EMBER dataset [1] with 800,000 annotated data samples extracted from real malware Windows executable files. For concept learning, RDF datasets of different sizes were curated based on the PE Malware Ontology [32]. The learning problem is to recognize the malware, i.e., to learn DL concepts that are true for all malware samples and false for all benign. Even black-box ML methods were not able to achieve

⁴ And parallel learning algorithms which are out of the focus of this report.

Table 1: `dataset_8_1000.owl` ($avg \pm std$).

Algorithm	Accuracy	Precision	Recall	FP Rate	F1
OCEL	0.75 \pm 0.02	0.77 \pm 0.03	0.72 \pm 0.04	0.22 \pm 0.05	0.74 \pm 0.02
CELOE	0.70 \pm 0.02	0.71 \pm 0.05	0.68 \pm 0.10	0.27 \pm 0.11	0.69 \pm 0.03
ELTL	0.65 \pm 0.03	0.60 \pm 0.02	0.93 \pm 0.02	0.61 \pm 0.05	0.73 \pm 0.02
DELTL	0.64 \pm 0.03	0.70 \pm 0.05	0.48 \pm 0.03	0.20 \pm 0.03	0.57 \pm 0.03
SPELL	0.71 \pm 0.03	0.74 \pm 0.04	0.64 \pm 0.04	0.22 \pm 0.04	0.69 \pm 0.03
ALC-SAT	0.71 \pm 0.02	0.73 \pm 0.05	0.67 \pm 0.03	0.25 \pm 0.06	0.70 \pm 0.02

Table 2: `dataset_1_10000.owl` ($avg \pm std$).

Algorithm	Accuracy	Precision	Recall	FP Rate	F1
OCEL	0.72 \pm 0.00	0.70 \pm 0.00	0.76 \pm 0.01	0.31 \pm 0.01	0.73 \pm 0.00
CELOE	0.71 \pm 0.00	0.66 \pm 0.01	0.84 \pm 0.00	0.42 \pm 0.00	0.74 \pm 0.00
ELTL	0.67 \pm 0.01	0.61 \pm 0.00	0.93 \pm 0.00	0.58 \pm 0.02	0.74 \pm 0.00
DELTL	0.63 \pm 0.00	0.71 \pm 0.01	0.45 \pm 0.00	0.18 \pm 0.01	0.55 \pm 0.00
SPELL	0.69 \pm 0.01	0.73 \pm 0.01	0.60 \pm 0.01	0.23 \pm 0.02	0.66 \pm 0.01
ALC-SAT	0.68 \pm 0.01	0.68 \pm 0.05	0.72 \pm 0.12	0.35 \pm 0.13	0.69 \pm 0.03

Table 3: `wannacry_1_20000.owl` ($avg \pm std$).

Algorithm	Accuracy	Precision	Recall	FP Rate	F1
OCEL	0.94 \pm 0.00	0.99 \pm 0.00	0.76 \pm 0.01	0.00 \pm 0.00	0.86 \pm 0.00
CELOE	0.83 \pm 0.00	0.62 \pm 0.02	0.83 \pm 0.07	0.17 \pm 0.03	0.86 \pm 0.00
ELTL	0.83 \pm 0.00	0.60 \pm 0.00	0.97 \pm 0.00	0.21 \pm 0.00	0.74 \pm 0.00
DELTL	0.80 \pm 0.00	0.92 \pm 0.00	0.24 \pm 0.01	0.00 \pm 0.00	0.38 \pm 0.01
SPELL	0.87 \pm 0.00	0.95 \pm 0.00	0.53 \pm 0.01	0.01 \pm 0.00	0.68 \pm 0.01
ALC-SAT	0.86 \pm 0.00	0.88 \pm 0.01	0.53 \pm 0.01	0.02 \pm 0.00	0.66 \pm 0.01

100% detection⁵ – and so there is, in fact, no exact target concept to be learned. Instead, standard ML methodology [10] is used to evaluate the quality of the learned concepts’ approximation of the sample in terms of accuracy, precision, recall, false positive (FP) rate, and F1 measure.

It is not feasible to process full EMBER, so we rely on selected partial datasets: `dataset_8_1000` contains 500+500 randomly selected malware and benign samples; `dataset_1_10000` has 5,000+5,000; and, differently, `wannacry_1_20000` contains 5,000 malware samples, all from the WannaCry malware family and 15,000 benign samples – this enables to find more precise output concepts, as samples from the same family are structurally similar [30, 32].

The experiments were run on a system with an 18-core Intel Core i9-10980XE processor, 256 GB of RAM, running Debian 5.10-140.1. The runtime limit was two hours. Evaluation used a 5-fold cross-validation [7] to reduce variance compared to a single training/validation split.

4 Results

The success measures of all runs are summarized in Tables 1–3; averages across the 5 folds are given. For better comparison, also the learning curves for accuracy

⁵ For instance, Oyama et al. [23], studying the most relevant combination of EMBER features, reported the accuracy of 92.7% in the case when all features were used.

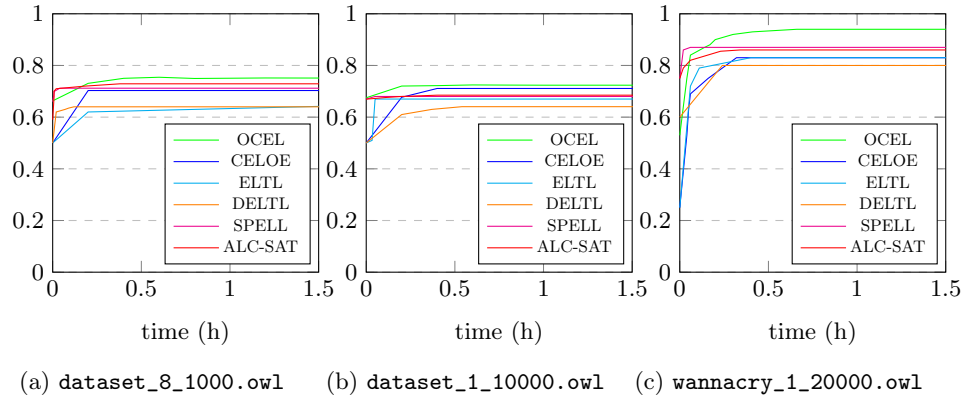


Fig. 1: Learning progression of individual algorithms (accuracy over time)

are printed in Figure 1. We remark that the expressive algorithms OCEL and CELOE were mainly included as a “more powerful” baseline – they allow more expressivity, hence they reach better approximation (in terms of accuracy and F1) – especially OCEL (for CELOE this is visible in the case of the 10k dataset). This is at the expense of the required time (the learning curves are less steep before flattening out). In contrast, SPELL and ALC-SAT are clearly progressing much faster than the remaining algorithms (the learning curves way are steeper).

It is very interesting to compare the actual learned concepts. Due to limited space, we may only show the most interesting concepts from the Wannacry dataset, selected among the different folds. We specifically picked the Wannacry family because here CELOE learned an \mathcal{EL} concept with high F1. We were naturally curious how the \mathcal{EL} and \mathcal{ELU} learners would fare.

```
(∃has_action.CreateFile) ⊓
(∃has_action.ProcessHandling) ⊓
(∃has_file_feature.PathStrings) ⊓
(≤ 1 has_action(BindAddressToSocket ⊓ CreateFile)                               (OCEL)
 ⊓ CreateFileSymbolicLink ⊓ DeleteFile
 ⊓ EnumerateRegistryKeyValues ⊓ GetSystemTime
 ⊓ SetProcessEnvironmentVariable ⊓ WindowHandling))

(∃has_action.CreateFile) ⊓ (∃has_file_feature.PathStrings)                       (CELOE)

(∃has_file_feature.NonstandardMZ) ⊓
(∃has_file_feature.PathStrings) ⊓
(∃has_section.InitializedDataSection) ⊓
(∃has_section.(CodeSection ⊓ (∃has_section_flag.Executable)
 ⊓ (∃has_section_flag.Readable))) ⊓
(∃has_section.(has_section_flag.Writable))                                     (ELTL)
```

$(\exists \text{has_action.CreateService}) \sqcap (\exists \text{has_action.GetFunctionAddress}) \sqcap$
 $(\exists \text{has_action.OpenCriticalSection}) \sqcap (\exists \text{has_action.ReadFromFile}) \sqcap$ (DELTL)
 $(\exists \text{has_action.StartService})$

DynamicLinkLibrary $\sqcap (\exists \text{has_action.CreateProcess})$
 $\sqcap (\exists \text{has_action.WriteToFile})$ (SPELL)
 $\sqcap (\exists \text{has_file_feature.NonstandardMZ})$
 $\sqcap (\exists \text{has_file_feature.PathStrings})$

$(\exists \text{has_action.CreateProcess}) \sqcap$ (ALC-SATa)
 $(\text{CloseRegistryKey} \sqcup \text{DynamicLinkLibrary})$

DynamicLinkLibrary \sqcap (ALC-SATb)
 $(\exists \text{has_action.}(\text{CreateProcess} \sqcup \text{SetSystemTime}))$

Our study confirmed that particularly OCEL reached concepts with higher expressivity, allowing for better approximation of the sample in EMBER, though they are also harder to interpret by humans (notably, the \leq restriction).

Concepts learned by the \mathcal{EL} and \mathcal{ELU} learners are more interpretable; however, their performance is mixed. ELTL was able to reach a higher F1 than SPELL or even ALC-SAT. This was due to a really high recall; on the other hand, it jointly had a higher FP rate. . . So, there is no free lunch, so to say. Also, the found ELTL concept contains $\exists \text{has_section.}(\text{CodeSection} \sqcap (\exists \text{has_section_flag.Executable})) \sqcap (\exists \text{has_section_flag.Readable})$, equivalent to the shorter $\exists \text{has_section.} \exists \text{has_section_feature.} \text{WriteExecuteSection}$ thanks to the ontology – which clearly is suboptimal.

Surprisingly, DELTL reached lower accuracy and F1 than ELTL, and in the case of Wannacry, it only yielded \mathcal{EL} concepts, although it learned in a more expressive language including disjunction. ALC-SAT learned \mathcal{ELU} concepts with significantly higher F1 than DELTL.

Altogether, our experiments show that both SPELL and ALC-SAT perform computationally very well on large real-world datasets. In terms of learned concepts, there are notable differences – but many are due to design – e.g. as noted before, even CELOE found an \mathcal{EL} concept with high F1, while SPELL and ALC-SAT do not find it, as they optimize for accuracy rather than F1. The obtained concepts will be useful for tuning SPELL and ALC-SAT in the future.

Another observation is that going from \mathcal{EL} to \mathcal{ELU} does not seem to help much on the EMBER dataset, at least not on the selected datasets. We plan to investigate this further. While none of the learned concepts constitutes a breakthrough finding in malware analysis, many of the features identified within the found concepts indicate possible malicious behavior to a high degree (incl. write-execute sections, non-standard MZ headers, process spawning, etc.). Above all, all learned concepts are immediately well interpretable to malware experts, which justifies the methodology and it is a significant step forward, compared to black-box ML

Acknowledgments. The coauthors wish to thank Jean Christoph Jung. Funded by the EU NextGenerationEU through the Recovery and Resilience Plan for Slovakia under the project No. 09I05-03-V02-00064.

References

1. Anderson, H.S., Roth, P.: EMBER: an open dataset for training static PE malware machine learning models. CoRR **abs/1804.04637** (2018), <http://arxiv.org/abs/1804.04637>
2. Baader, F., Brandt, S., Lutz, C.: Pushing the \mathcal{EL} envelope. In: Kaelbling, L.P., Saffiotti, A. (eds.) IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30 - August 5, 2005. pp. 364–369 (2005), <http://ijcai.org/Proceedings/05/Papers/0372.pdf>
3. Bühmann, L., Lehmann, J., Westphal, P.: DL-learner - A framework for inductive learning on the semantic web. J. Web Semant. **39**, 15–24 (2016). <https://doi.org/10.1016/J.WEBSEM.2016.06.001>
4. ten Cate, B., Funk, M., Jung, J.C., Lutz, C.: SAT-based PAC learning of description logic concepts. In: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China. pp. 3347–3355. ijcai.org (2023). <https://doi.org/10.24963/IJCAI.2023/373>
5. Darwiche, A.: Three modern roles for logic in AI. In: Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2020, Portland, OR, USA, June 14-19, 2020. pp. 229–243. ACM (2020). <https://doi.org/10.1145/3375395.3389131>
6. Darwiche, A.: Logic for explainable AI. In: 38th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2023, Boston, MA, USA, June 26-29, 2023. pp. 1–11. IEEE (2023). <https://doi.org/10.1109/LICS56636.2023.10175757>
7. Diamantidis, N.A., Karlis, D., Giakoumakis, E.A.: Unsupervised stratification of cross-validation for accuracy estimation. Artif. Intell. **116**(1-2), 1–16 (2000). [https://doi.org/10.1016/S0004-3702\(99\)00094-6](https://doi.org/10.1016/S0004-3702(99)00094-6)
8. Dolejš, J., Jureček, M.: Interpretability of machine learning-based results of malware detection using a set of rules. In: Artificial Intelligence for Cybersecurity, pp. 107–136. Springer International Publishing (2022). https://doi.org/10.1007/978-3-030-97087-1_5
9. Fanizzi, N., d’Amato, C., Esposito, F.: DL-FOIL concept learning in description logics. In: Inductive Logic Programming, 18th International Conference, ILP 2008, Prague, Czech Republic, September 10-12, 2008, Proceedings. LNCS, vol. 5194, pp. 107–121. Springer (2008). https://doi.org/10.1007/978-3-540-85928-4_12
10. Fawcett, T.: An introduction to ROC analysis. Pattern recognition letters **27**(8), 861–874 (2006)
11. Funk, M., Jung, J.C., Voellmer, T.: SAT-based bounded fitting for the description logic \mathcal{ALC} (extended abstract). In: 38th International Workshop On Description Logics, Opole, Poland (2025)
12. 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)
13. Haase, C., Lutz, C.: Complexity of subsumption in the \mathcal{EL} family of description logics: Acyclic and cyclic tboxes. In: Ghallab, M., Spyropoulos, C.D., Fakotakis, N., Avouris, N.M. (eds.) ECAI 2008 - 18th European Conference on Artificial

- Intelligence, Patras, Greece, July 21-25, 2008, Proceedings. *Frontiers in Artificial Intelligence and Applications*, vol. 178, pp. 25–29. IOS Press (2008). <https://doi.org/10.3233/978-1-58603-891-5-25>
14. Harang, R.E., Rudd, E.M.: SOREL-20M: A large scale benchmark dataset for malicious PE detection. *CoRR abs/2012.07634* (2020), <https://arxiv.org/abs/2012.07634>
 15. Hinton, G.E.: Connectionist learning procedures. *Artif. Intell.* **40**(1-3), 185–234 (1989). [https://doi.org/10.1016/0004-3702\(89\)90049-0](https://doi.org/10.1016/0004-3702(89)90049-0)
 16. Horrocks, I., Sattler, U.: A tableaux decision procedure for SHOIQ. In: *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, Edinburgh, Scotland, UK, July 30 - August 5, 2005. pp. 448–453 (2005), <http://ijcai.org/Proceedings/05/Papers/0759.pdf>
 17. Lehmann, J.: *Learning OWL Class Expressions, Studies on the Semantic Web*, vol. 6. IOS Press, Amsterdam, Netherlands (2010). <https://doi.org/10.3233/978-1-61499-340-7-i>
 18. Lehmann, J., Haase, C.: Ideal downward refinement in the EL description logic. In: De Raedt, L. (ed.) *Inductive Logic Programming, 19th International Conference. Lecture Notes in Computer Science*, vol. 5989, pp. 73–87. Springer (2009). https://doi.org/10.1007/978-3-642-13840-9_8
 19. Lehmann, J., Hitzler, P.: Concept learning in description logics using refinement operators. *Mach. Learn.* **78**(1-2), 203–250 (2010). <https://doi.org/10.1007/S10994-009-5146-2>
 20. Lundberg, S.M., Lee, S.: A unified approach to interpreting model predictions. In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. pp. 4765–4774 (2017). <https://doi.org/10.48550/arXiv.1705.07874>
 21. Marais, B., Quertier, T., Chesneau, C.: Malware analysis with artificial intelligence and a particular attention on results interpretability. In: *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
 22. 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 (2019)
 23. Oyama, Y., Miyashita, T., Kokubo, H.: Identifying useful features for malware detection in the ember dataset. In: *Seventh International Symposium on Computing and Networking Workshops, CANDAR 2019 Workshops, Nagasaki, Japan, November 26-29, 2019*. pp. 360–366. IEEE (2019). <https://doi.org/10.1109/CANDARW.2019.00069>
 24. Ribeiro, M.T., Singh, S., Guestrin, C.: "Why should I trust you?": Explaining the predictions of any classifier. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. pp. 1135–1144. ACM (2016). <https://doi.org/10.1145/2939672.2939778>
 25. Rizzo, G., Fanizzi, N., d'Amato, C.: Class expression induction as concept space exploration: From dl-foil to dl-focl. *Future Gener. Comput. Syst.* **108**, 256–272 (2020). <https://doi.org/10.1016/J.FUTURE.2020.02.071>
 26. Schwalbe, G., Finzel, B.: A comprehensive taxonomy for explainable artificial intelligence: a systematic survey of surveys on methods and concepts. *Data Min. Knowl. Discov.* **38**(5), 3043–3101 (2024). <https://doi.org/10.1007/S10618-022-00867-8>

27. 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)
28. Ucci, D., Aniello, L., Baldoni, R.: Survey of machine learning techniques for malware analysis. *Computers & Security* **81**, 123–147 (2019). <https://doi.org/10.1016/j.cose.2018.11.001>
29. Valiant, L.G.: A theory of the learnable. *Commun. ACM* **27**(11), 1134–1142 (1984). <https://doi.org/10.1145/1968.1972>
30. Švec, P.: Ontologická reprezentácia pre bezpečnosť informačných systémov [Ontological representation for security of information systems]. Phd thesis, Slovak University of Technology in Bratislava, Faculty of Electrical Engineering and Informatics, Bratislava, Slovakia (2024), https://uim.fei.stuba.sk/wp-content/uploads/2024/10/2024.svec_.dizp_.pdf
31. 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). <https://doi.org/10.3233/SW-180308>
32. Švec, P., Balogh, S., Homola, M., Kluka, J., Bisták, T.: Semantic data representation for explainable windows malware detection models. *CoRR* **abs/2403.11669** (2024). <https://doi.org/10.48550/ARXIV.2403.11669>