Compiling Neural Network Classifiers into Boolean Circuits for Efficient Shap-Score Computation

Leopoldo Bertossi[1], Jorge E. León[2]

[1]SKEMA Business School, Montreal, Canada
[2]Universidad Adolfo Ibáñez (UAI), Santiago, Chile

Abstract

We describe the transformation of binary neural networks (BNNs) for classification into deterministic and decomposable Boolean circuits by means of knowledge compilation techniques. The resulting circuit is used, as an open-box model, to compute Shap scores taking advantage of a recent efficient algorithm for Shap on this class of circuits. We show experimental results that corroborate the huge improvement in score computation time in comparison with the computation that directly uses the BNN as a black-box model.

Keywords

Explainable ML, Shap Scores, Knowledge Compilation, Neural Networks, Boolean Decision Circuits

1. Introduction

Explanations for the outcomes from classification models come in different forms, and can be obtained through different approaches. A common one assigns *attribution scores* to the features values associated to an input that goes through an ML-based model, to *quantify* their relevance for the obtained outcome. We concentrate on *local* scores, i.e. associated to a particular input, as opposed to global scores that indicates the overall relevance of a feature. In this work, we also concentrate on explanations for binary classification models, whose features take binary values, so as the classification label, say 0 or 1.

A popular local score is Shap [10], which is based on the Shapley value that has introduced and used in coalition game theory and practice for a long time [15, 13]. Shap scores can be computed with a black-box or an open-box model [14]. With the former, we do not know or use its internal components, but only its input/output relation. This is the most common approach. In the latter case, we can have access to its internal structure and components, and we can use them for score computation. It is common to consider neural-network-based models as black-box models, because their internal gates and structure may be difficult to understand or process when it comes to explaining classification outputs. However, a decision-tree model, due to its much simpler structure and use, is considered to be open-box for the same purpose.

Even for binary classification models, the complexity of Shap computation is provably hard, actually $\#P$-hard for several kinds of binary classification models, independently from whether

AMW 2023: Alberto Mendelzon International Workshop on Foundations of Data Management, May 22–26, 2023, Santiago, Chile

leopoldo.bertossi@skema.edu (L. Bertossi); jorgleon@alumnos.uai.cl (J. E. León)

the internal components of the model are used when computing Shap [3, 1, 2]. However, there are classes of classifiers for which, using the model components and structure, the complexity of Shap computation can be brought down to polynomial time [11, 2, 18].

A polynomial time algorithm for Shap computation with *deterministic and decomposable Boolean circuits* (dDBCs) was presented in [2]. From this result, the tractability of Shap computation can be obtained for a variety of Boolean circuit-based classifiers than can be represented as (or compiled into) them. In particular, this holds for *Ordered Binary Decision Diagrams* (OBDDs) [5], decision trees, and other established classification models that can be compiled into OBDDs [16, 6, 12]. Similar results can be obtained for *Sentential Decision Diagrams* (SDDs) [9], which can be seen as dDBCs, and form a convenient *knowledge compilation* target language [7, 17]. In [18], through a different approach, tractability of Shap computation was obtained for a collection of classifiers that intersect with that in [2].

In this work, we concentrate on explicitly developing this approach to the efficient computation of Shap for *binary neural networks* (BNNs). For this, and inspired by [16], a BNN is transformed into a dDBC using techniques from *knowledge compilation* [7], an area that investigates the transformation of (usually) propositional theories into an equivalent one with a canonical syntactic form that has some good computational properties, e.g. tractable model counting. The compilation may incur in a relatively high computational cost [7, 8], but it may still be worth the effort when a particular property will be checked often, as is the case of explanations for the same BNN.

We also make experimental comparisons between this open-box and circuit-based Shap computation and that based directly on the BNN treated as a black-box, i.e. using only its input/output relation. We perform comparisons in terms of computation time and alignment of Shap scores. We confirm that Shap computation via the dDBC vastly outperforms the direct Shap computation on the BNN. It is also the case that the scores obtained are fully aligned, as expected since the dDBC represents the BNN. A detailed account of our work can be found in [4].

2. Background

Consider a fixed entity $\mathbf{e} = \langle F_1(\mathbf{e}), \ldots, F_N(\mathbf{e})\rangle$ subject to classification. It has values $F_i(\mathbf{e})$ for features $\mathcal{F} = \{F_1, \ldots, F_N\}$. In [10, 11], the Shapley value is applied with $\{F_1(\mathbf{e}), \ldots, F_N(\mathbf{e})\}$ as the set of players, and the game function $\mathcal{G}_{\mathbf{e}}(s) := \mathbb{E}(L(\mathbf{e}') \mid \mathbf{e}'_s = \mathbf{e}_s)$, giving rise to the Shap score. Here, $\mathbf{e}_s$ is the projection (or restriction) of $\mathbf{e}$ on (to) the subset s of features, L is the label. The $\mathbf{e}'$ inside the expected value is an entity whose values coincides with those of $\mathbf{e}$ for the features in s. For $F \in \mathcal{F}$, and entity $\mathbf{e}$:

$$\begin{aligned}\mathsf{Shap}(\mathcal{F}, \mathcal{G}_{\mathbf{e}}, F) \;=\; & \sum_{s \subseteq \mathcal{F}\setminus\{F\}} \frac{|s|!(|\mathcal{F}| - |s| - 1)!}{|\mathcal{F}|!} \times \\ & \qquad [\, \mathbb{E}(L(\mathbf{e}') \mid \mathbf{e}'_{S\cup\{F\}} = \mathbf{e}_{s\cup\{F\}}) - \mathbb{E}(L(\mathbf{e}') \mid \mathbf{e}'_s = \mathbf{e}_s)\,].\end{aligned}$$

The expected value is defined on the basis of an underlying probability distribution on the entity population. Shap quantifies the contribution of feature value $F(\mathbf{e})$ to the outcome label.

In order to compute Shap, we only need function L, and none of the internal components of the classifier. Given that all possible subsets of features appear in its definition, Shap is bound to be hard to compute. Actually, for some classifiers, its computation may become $\#P$-hard (c.f. [2] for some cases). However, in [2], it is shown that Shap can be computed in polynomial time for every *deterministic and decomposable Boolean circuit* (dDBC) used as a classifier. The circuit's internal structure is used in the computation.

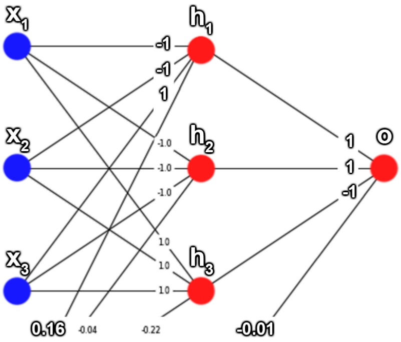
3. BNN Compilation

A BNN is first compiled into a propositional formula in Conjunctive Normal Form (CNF), which, in its turn, is compiled into an SDD, which is finally compiled into a dDBC. We compute Shap on the resulting circuit via the efficient algorithm in [2]. This compilation is performed once, and is independent from any input to the classifier. The final circuit can be used to compute Shap scores for different input entities. We show the compilation path by means of a simple example.

In our BNNs we used, for each gate g, a *step function* as activation function, of the form:

$$\phi_g(\bar{i}) = sp(\bar{w}_g \bullet \bar{i} + b_g) \ := \begin{cases} 1 & \text{if } \bar{w}_g \bullet \bar{i} + b_g \geq 0, \\ -1 & \text{otherwise,} \end{cases}$$

which is parameterized by a vector of binary weights $\bar{w}_g$ and a real-valued constant bias b_g. For technical, non-essential reasons, we used inputs and outputs, $-1, 1$, except for the output gate, o, that returns 0 or 1.

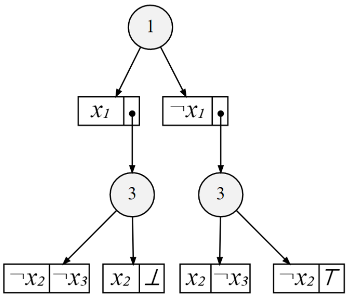


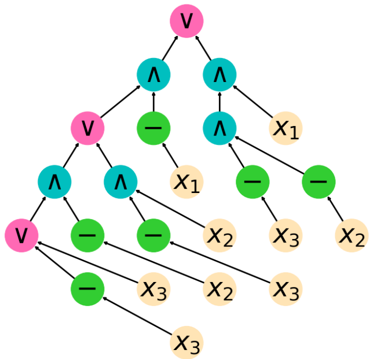
$$\begin{aligned} o \longleftrightarrow\ & (-[(x_3 \wedge (x_2 \vee x_1)) \vee (x_2 \wedge x_1)] \wedge \\ & ([(-x_3 \wedge (-x_2 \vee -x_1)) \vee (-x_2 \wedge -x_1)] \vee \\ & [(x_3 \wedge (-x_2 \vee -x_1)) \vee (-x_2 \wedge -x_1)])) \vee \\ & ([(-x_3 \wedge (-x_2 \vee -x_1)) \vee (-x_2 \wedge -x_1)] \wedge \\ & [(x_3 \wedge (-x_2 \vee -x_1)) \vee (-x_2 \wedge -x_1)]). \end{aligned}$$

In CNF:

$$o \longleftrightarrow (-x_1 \vee -x_2) \wedge (-x_1 \vee -x_3) \wedge (-x_2 \vee -x_3).$$

The BNN on the LHS above is transformed into the propositional formula on the RHS. After that, it is transformed into a simplified CNF formula, which is shown right below. In its turn, the CNF is transformed into the SDD on the RHS below. In general, this is the most expensive step during the overall compilation time, but it has an upper bound that is exponential in the tree-width of the formula [9].





Finally, the SDD is easily transformed into a dDBC, the one shown on the RHS here. It can be used as a binary classifier, with binary input features x_1, x_2, x_3. The binary label is read off from the top node. This circuit is *deterministic* in that, for every $\vee$-gate, at most one of its inputs is 1 when the output is 1. It is *decomposable* in that, for every $\wedge$-gate, the inputs do not share features. This dDBC is also *smooth*, in that sub-circuits that feed a same $\vee$-gate share the same features. It has a *fan-in* at most two, in that every $\wedge$-gate and $\vee$-gate have at most two inputs.

4. Experiments

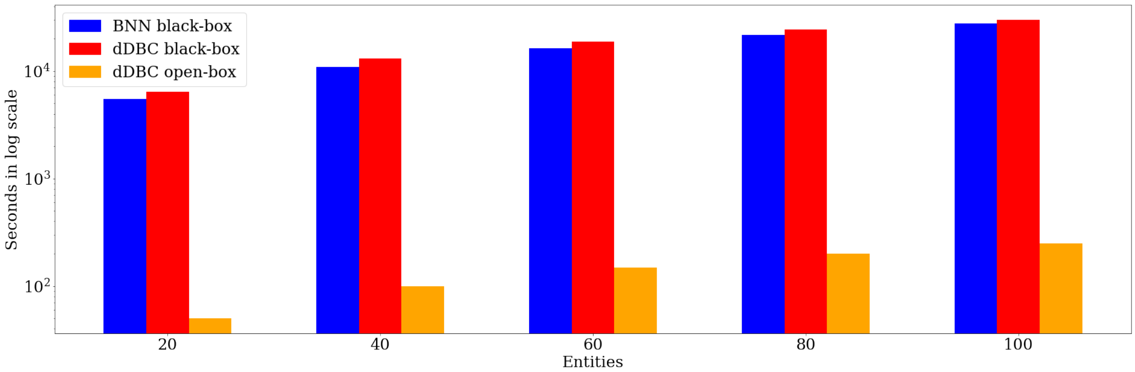
For our experiments, we consider real estate as an application domain, where house prices depend on certain features, which we appropriately binarize.[1] The problem consists in classifying property blocks, represented as entity records of thirteen feature values, as *high-value* or *low-value*. This is a binary classification problem for which a BNN is first learnt, and then used.

After the transformation of the BNN into circuit $\mathcal{C}$, we had the possibility to compute Shap, for a given input entity, in three different ways: (a) Directly on the BNN as a black-box model, i.e. using only its input/output relation for multiple calls; (b) Similarly, using the circuit $\mathcal{C}$ as a black-box model; and (c) With the efficient algorithm in [2] for smooth dDBCs with fan-in 2, treating circuit $\mathcal{C}$ as an open-box.

We performed these three computations for sets of 20, 40, 60, 80, and 100 input entities, to compare average times with increasing numbers of entities. In all cases, Shap was computed with the uniform probability distribution over the joint feature domain of size 2^{13}. The results shown right below report on the seconds taken to compute Shap on 20, 40, 60, 80 and 100 entities; using the BNN as a black-box (blue bar), the dDBC as a black-box (red bar), and the dDBC as an open-box (orange bar). Note that the vertical axis employs a logarithmic scale.

In our experiment, the initial transformation of the BNN into CNF took 1.3 hrs, which was the most expensive paper of the compilation. However, it is a one time computation, and our rather naive transformation algorithm leaves considerable room for improvement.

[1] We use the California Housing Prices dataset available at https://www.kaggle.com/datasets/camnugent/california-housing-prices.



5. Acknowledgments

We are grateful, for their scientific and technical help, to: Arthur Choi, Andy Shih, Norbert Manthey, Maximilian Schleich, and Adnan Darwiche. Part of this work was funded by ANID - Millennium Science Initiative Program - Code ICN17002; and "Centro Nacional de Inteligencia Artificial" CENIA, FB210017 (Financiamiento Basal para Centros Científicos y Tecnológicos de Excelencia de ANID), Chile. Both CENIA and SKEMA Canada funded Jorge León's visit to Montreal. L. Bertossi is a Professor Emeritus at Carleton University, Ottawa, Canada; and a Senior Fellow of the Universidad Adolfo Ibáñez (UAI), Chile.

References

[1] Arenas, M., Barceló, P., Bertossi, L. and Monet, M. On the Complexity of SHAP-Score-Based Explanations: Tractability via Knowledge Compilation and Non-Approximability Results. *Journal of Machine Learning Research*, 2023, 24(63):1-58. Extended version of [2].

[2] Arenas, M., Barceló, P., Bertossi, L. and Monet, M. The Tractability of SHAP-Score-Based Explanations for Classification over Deterministic and Decomposable Boolean Circuits. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, 2021, 6670–6678.

[3] Bertossi, L., Li, J., Schleich, M., Suciu, D. and Vagena, Z. Causality-Based Explanation of Classification Outcomes. In *Proceedings of the 4th International Workshop on "Data Management for End-to-End Machine Learning" (DEEM) at ACM SIGMOD/PODS*, 2020, 1–10. Posted as ArXiv paper 2003.06868.

[4] Bertossi, L. and Leon, J. E. Opening Up the Neural Network Classifier for Shap Score Computation. 2023, ArXiv paper 2303.06516.

[5] Bryant, R. E. Graph-Based Algorithms for Boolean Function Manipulation. *IEEE Transactions on Computers*, 1986, C-35(8):677–691.

[6] Darwiche, A. and Hirth, A. On The Reasons Behind Decisions. In *Proceedings of the 24th European Conference on Artificial Intelligence*, 2020, 712–720.

[7] Darwiche, A. and Marquis, P. A Knowledge Compilation Map. *Journal of Artificial Intelligence Research*, 2002, 17(1):229–264.

[8] Darwiche, A. On the Tractable Counting of Theory Models and its Application to Truth Maintenance and Belief Revision. *Journal of Applied Non-Classical Logics*, 2011, 11(1-2):11–34.

[9] Darwiche, A. SDD: A New Canonical Representation of Propositional Knowledge Bases. In *Proceedings of the 22th International Joint Conference on Artificial Intelligence (IJCAI-11)*, 2011, 819–826.

[10] Lundberg, S. M. and Lee, S.-I. A Unified Approach to Interpreting Model Predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, 4768–4777. ArXiv paper 1705.07874.

[11] Lundberg, S., Erion, G., Chen, H., DeGrave, A., Prutkin, J., Nair, B., Katz, R., Himmelfarb, J., Bansal, N. and Lee, S.-I. From Local Explanations to Global Understanding with Explainable AI for Trees. *Nature Machine Intelligence*,2020, 2(1):56–67. ArXiv paper 1905.04610.

[12] Narodytska, N., Kasiviswanathan, S., Ryzhyk, L., Sagiv, M. and Walsh, T. Verifying Properties of Binarized Deep Neural Networks. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018, 6615–6624.

[13] Roth, A. E. *The Shapley Value: Essays in Honor of Lloyd S. Shapley.* Cambridge University Press, 1988.

[14] Rudin, C. Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead. *Nature Machine Intelligence*, 2019, 1:206–215. ArXiv paper 1811.10154.

[15] Shapley, L. S. A Value for n-Person Games. In *Contributions to the Theory of Games (AM-28)*, 1953, vol. 2, 307–318.

[16] Shi, W., Shih, A., Darwiche, A. and Choi, A. On Tractable Representations of Binary Neural Networks. In *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning*, 2020, 882–892. ArXiv paper 2004.02082.

[17] Van den Broeck, G. and Darwiche, A. On the Role of Canonicity in Knowledge Compilation. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, 2015, 1641–1648.

[18] Van den Broeck, G., Lykov, A., Schleich, M. and Suciu, D. On the Tractability of SHAP Explanations. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, 2021, 6505–6513.