Boolean Expression Approximation for Binary Classification Using BBQT

Alexander Stahl[1]

[1]Brandenburg University of Technology, Cottbus, Germany

Abstract

To this day, there exist no specific induction algorithms for Commuting Quantum Query Logic (CQQL) conditions. In this work we explore the possibilities of using the training method of BBQ-Trees as a means to approximate hidden CQQL conditions. An experimental study showcases the applicability of the approach.

Keywords

classifier, interpretable AI, decision tree, sample data analysis, logic induction

1. Formula Reconstruction

1.1. Problem Definition

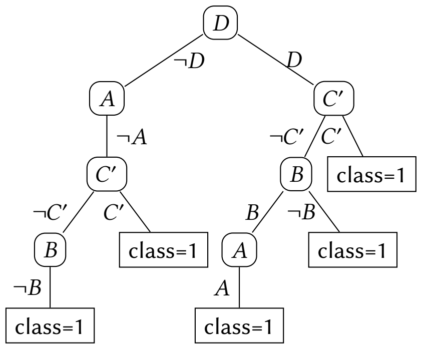
Commuting Quantum Query Language (CQQL) [1] is a quantum-logic-inspired language that syntactically corresponds to Boolean expressions, utilizing conjunction, disjunction and negation, while regarding truth values as gradual within the range [0,1]. Here, a higher value indicates greater fulfillment of the expressed condition. Unlike fuzzy logic, CQQL adheres to all the rules of Boolean algebra. One of its notable applications is the BBQ-Tree (BBQT) [2], which constitutes a threshold-based binary classification model that represents a learned CQQL condition combined with a well-chosen global threshold. This allows for interpretable and accurate binary classification based on the gradual truth values inherent in CQQL.

So far, CQQL conditions are typically specifically crafted by a user or a system based on user-preferences. Instead, this paper addresses the challenge of uncovering unknown conditions from data. We assume the existence of a hidden CQQL expression represented as a BBQ-Tree and discuss the recovery of the original expression through the BBQT training algorithm. While the recovery of classical Boolean expressions or values can be achieved using truth tables, here the process is more complex. This paper aims to explore and test methods for effectively retrieving these hidden conditions.

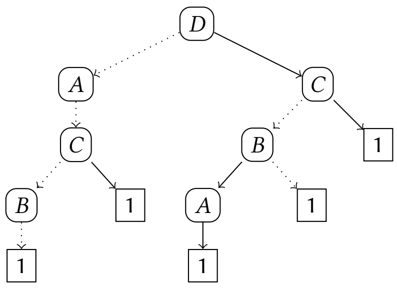
1.2. Related Works

To this day, specific induction algorithms for CQQL conditions do not exist. Certain edge cases have been considered, such as in Schmitt and Zellhöfer's work [3], which explores the learning of CQQL conditions from user preferences. However, this approach differs from our objective, as it requires preferences to be pre-modeled or explicitly expressed. A related concept is Fuzzy Logic (FL), whose induction can be achieved through various methods. For instance, optimal FL conditions can be generated using genetic algorithms, as demonstrated in studies such as the ones by Angelov [4], Wen-yuan [5] or Linkens [6]. Similar to classical decision trees, Fuzzy Decision Trees [7], which are modified to handle FL operations, can be trained and utilized to determine conditions. Additionally, it is possible to extract FL conditions from the results of specialized clustering methods, such as c-means [8]. Ultimately, these FL methods focus on learning FL rather than CQQL, which bears the disadvantage of noncompliance to important logic laws such as idempotence or the law of excluded middle [1].

LWDA'24: Lernen, Wissen, Daten, Analysen. September 23–25, 2024, Würzburg, Germany
stahlale@b-tu.de (A. Stahl)
0009-0009-0554-107X (A. Stahl)
© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).



(a) Original Boolean Expression as a BBQT ($C' \equiv (C > 0.3)$)



(b) Reconstructed BBQT

Figure 1: Reconstruction or Exemplary BBQT (#13)

Generally speaking, the use of established optimization methods such as genetic algorithms for the determination of CQQL conditions has not yet been explored, which is why only training methods for BBQT are currently available for this purpose.

2. Induction of BBQTs

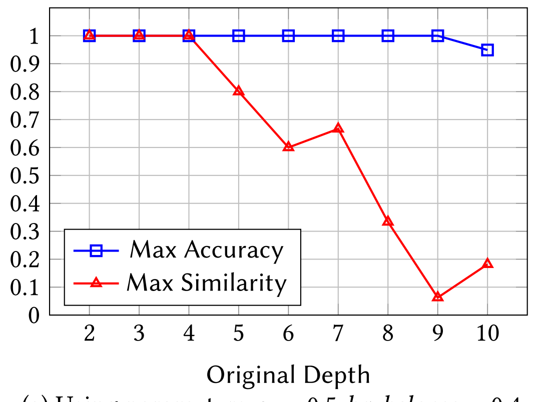
In order to explore the extraction of valid CQQL conditions from labeled data we assume that the hidden condition exists in the form of a BBQT. This also implies the existence of a threshold, as defined in this model. In the absence of a suitable threshold, 0.5 is possible as a neutral value. The applied methodology of this study is as follows: In our experiments, we are creating artificial datasets that are generated by the hidden BBQT on random training objects $o \in [0, 1]^n \subseteq \mathbb{R}^n$ with n attribute values. The resulting classification result $K(o) \in \{0, 1\}$ is treated as the label of the training object. This is then employed to examine how well the training algorithm presented in [2] can reconstruct the hidden BBQT based on derived labels. Consider the following example scenario to illustrate the experiment setup: We start by defining four attributes A, B, C, D as well as three mutually exclusive paths of a tree, connected via disjunction: $(\neg A \wedge \neg B) \vee (\neg A \wedge B \wedge (C > 0.3)) \vee (A \wedge D)$. This expression is visualized by the tree in Figure 1a. Note that the tree was reorganized for better visual comparability, while remaining logically equivalent to the mentioned Boolean expression.

We assumed $\tau = 0.5$ and used the tree to classify a total of 10,256 arbitrary objects, thus creating labeled data, from which $\mathbb{TR}$ and $\mathbb{TE}$ were chosen. During this procedure, it was ensured that the class distribution is balanced. Subsequently, the objects were used to train and test a set of classifiers (that did not have access to the original tree/expression). The results of this experiment are shown in Table 1. In the table, the configurations of various training runs are presented together with their respective achieved accuracy value. For an explanation of the different tree types and parameters, please consult [2]. Figure 1b depicts one possible resulting BBQT (result #13 in Table 1). Here it is notable that the inner nodes of the original and resulting BBQT differ slightly. In a BBQT, each node can be one of three different types of splits, which determines its behavior in the model evaluation. In the given Figure, C' marks a so-called B-Split that behaves like a classical split in a DT, whereas the other splits are Q-splits. A more in-depth description of these splits can be found in [2]. Noticable is that if one substitutes the original B-split node $C > 0.3$ with the Q-split node C then the logical expressions of the tree are equivalent to the target expressions defined above. Thus, we deem the reconstruction of the logical expression successful, with only one B-split (on C) misinterpreted as a Q-split. Even though the substitution of $C > 0.3$ with C reduces the accuracy from 100 % to 92 %, it can be stated that all derived trees are logically close to the given tree. Hence, we can assert for this example that the approach is able to reproduce the given BBQT.

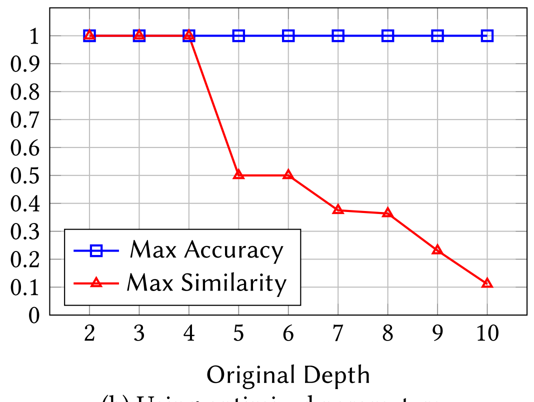
In order to better quantify the capability of the approach, we performed a variety of experiments that follow the same setup. In each experiment, we first generate a random BBQT and utilize it to

#	Type	Level	ρ_θ	*bq_balance*	Acc.	#	Type	Level	ρ_θ	*bq_balance*	Acc.
5	BQT	2	0.5	0.3	0.8	12	BQT	3	0.6	0.1	0.87
6	BQT	2	0.5	0.1	0.8	13	BQT	4	0.5	0.3	0.92
7	BQT	2	0.6	0.3	0.74	14	BBQT	3/4	0.5	0.3	0.9
8	BQT	2	0.6	0.1	0.74	15	BBQT	4/6	0.5	0.3	0.92
9	BQT	3	0.5	0.3	0.87	16	BQT+DT	2/4	0.5	0.3	0.93
10	BQT	3	0.5	0.1	0.87	17	BQT+DT	3/4	0.5	0.3	0.95
11	BQT	3	0.6	0.3	0.87	18	BQT+DT	4/4	0.5	0.3	0.97

Table 1
Results for BBQ formula reconstruction



(a) Using parameters $\rho_\theta = 0.5, bq\_balance = 0.4$



(b) Using optimized parameters

Figure 2: Maximum Values for Random BBQT Reconstruction

generate the label $K(o)$ for 1000 random objects o, ensuring that no duplicate objects are used. We then split these into a training and test set $\mathbb{TR}$ and $\mathbb{TE}$ with $|\mathbb{TR}| = |\mathbb{TE}| = 500$, which were balanced via undersampling. These are consequently applied to train a second BBQT. As the visual and manual comparison between the original and resulting BBQT is both error-prone and time-intensive, we instead opted to quantify their similarity. To this end, we utilized the fact that BBQTs correspond to a set of minterms in disjunctive normal form. We were thus able to calculate a similarity score between two BBQTs by using the Jaccard similarity coefficient on their sets of active minterms, as shown in Equation 1. Note that I_n is the set of active minterms of tree n.

$$sim(BBQ_1, BBQ_2) = \frac{\{m_i | i \in I_1\} \cap \{m_i | i \in I_2\}}{\{m_i | i \in I_1\} \cup \{m_i | i \in I_2\}} \tag{1}$$

Running this experiment 5000 times, we extracted the measured classification accuracy as well as the tree similarity, keeping track of the depth of the original tree. Proper tuning of the model's hyperparameters remains an area of ongoing research, which is why not every BBQT produced optimal results. Here we only depict the maximum values to showcase the theoretical possibilities of the approach. The results can be observed in Figure 2, where the maximum achieved accuracy and similarity is depicted in relation to the depth of the original tree. Figure 2a shows this data as a result of training a BBQT with the exemplary hyperparameters $\rho_\theta = 0.5$ and $bq\_balance = 0.4$. Figure 2b, on the other hand, shows the results of a BBQT trained with optimal hyperparameters, that were optimized for *max accuracy*. Obviously, is not viable to optimize the hyperparameters for maximum similarity, since the original BBQT is not known in any other but our experimental scenario. It can be seen that the depth of the BBQT and therefore the complexity of the underlying CQQL condition has a great impact on the recovery quality, if measured in similarity. Regarding accuracy, the resulting tree can be seen as adequate.

3. Conclusions

In this study, we demonstrated that the reconstruction of a hidden CQQL condition in the form of a BBQT is feasible when result data from the tree is available. Our experiments explored the performance of this approach using exemplary trees and datasets. BBQT training remains challenging due to the complexity of hyperparameter selection and its significant impact on the results. Our current training algorithms focus on optimizing parameters solely to maximize accuracy, since the original tree is hidden, meaning that the similarity calculations used in our experiments are not practical for real-world applications. To refine the approach, a possible variant of BBQTs that produces metric output (regression) could be more effective, as it would eliminate the need for a predefined temporary threshold. Alternatively, the original threshold could be treated as a hyperparameter, further enhancing the flexibility and applicability of the method.

References

[1] I. Schmitt, QQL: A DB&IR query language, The VLDB Journal 17 (2008) 39–56. doi:10.1007/s00778-007-0070-1.

[2] A. Stahl, I. Schmitt, BBQ-Tree – a decision tree with boolean and quantum logic decisions, in: Lecture Notes in Computer Science, volume 14918, 2024.

[3] I. Schmitt, D. Zellhöfer, Condition learning from user preferences, in: 2012 Sixth International Conference on Research Challenges in Information Science (RCIS), 2012.

[4] P. Angelov, R. Buswell, Automatic generation of fuzzy rule-based models from data by genetic algorithms, Information Sciences 150 (2003) 17–31. doi:10.1016/S0020-0255(02)00367-5, recent Advances in Soft Computing.

[5] L. Wen-yuan, X. Chun-jing, Fuzzy rule generation based on genetic algorithm, Computer Simulation (2005).

[6] D. Linkens, H. O. Nyongesa, Real-time acquisition of fuzzy rules using genetic algorithms, IFAC Proceedings Volumes 17 (1992) 335–339.

[7] C. Z. Janikow, Fuzzy decision trees: issues and methods, IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society 28 1 (1998) 1–14.

[8] J. C. Bezdek, R. Ehrlich, W. Full, Fcm: The fuzzy c-means clustering algorithm, Computers & Geosciences 10 (1984) 191–203. doi:10.1016/0098-3004(84)90020-7.