BRL: A Toolkit for Learning How an Agent Performs Belief Revision

Aaron Hunter and Konstantin Boyarinov
Department of Computing, BC Institute of Technology, Burnaby, Canada

Keywords: Belief Revision, Knowledge Representation, Learning.

Abstract: Belief revision occurs when an agent receives new information that may conflict with their current beliefs. This process can be modelled by a formal belief revision operator. However, in a practical scenario, simply defining abstract revision operators is not sufficient. A truly intelligent agent must be able to observe how others have revised their beliefs in the past, and use this information to predict how they will revise their beliefs in the future. In other words, an agent must be able to learn the mental model that is used by other agents. This process involves combining two traditionally distinct areas of Artificial Intelligence to produce a general reasoning system. In this paper, we discuss challenges faced in using various learning approaches to learn belief revision operators. We then present the BRL toolkit: software can learn the revision operator an agent is using based on past revisions. This is a tool that bridges formal reasoning and machine learning to address a common problem in practical reasoning. Accuracy and efficiency of the approach are discussed.

1 INTRODUCTION

Belief revision is the process in which an agent incorporates new information together with some pre-existing set of beliefs. This is important in any reasoning context where it is useful for an agent to develop a model of how other agents are likely to behave. There are many situations where we would like to predict the behaviour of an agent, based on past data. As an illustrative example, consider an agent that is controlling a motorized vehicle. We know the agent is rational, but we do not know exactly how the agent makes decisions about actions such as stopping and turning. We would like to be able to predict how this agent will respond to new information. For example, we may be interested in determining what it will do if a small animal appears in front of it. By looking at past revisions, we can try to learn if the presence of small animals triggers a change in belief. Note that we are not simply making predictions based on past actions; we are trying to learn the underlying revision operator so that we can accurately model the reasoning process. This will provide a more robust predictive model.

We are concerned with *learning* how an agent revises their beliefs, based on past information. Of course, there are many different approaches does not involve machine learning at all. We will discuss this approach, and provide a tool for automating the process. We then to consider the application of machine learning towards this task, considering some different basic models. We then introduce the Belief Revision Learner (BRL) toolkit; this is a software tool that uses past revision data to predict how an agent will revise their beliefs in the future.

2 PRELIMINARIES

2.1 Belief Revision

We introduce belief revision operators. These are formal operators defined in the setting of propositional logic. We therefore assume an underlying *vocabulary* V is a finite set of propositional variables. A *formula* is a propositional combination of symbols in V, formed using the usual connectives. A *state* is a propositional interpretation of V; in other words, a state assigns true-false values to every variable. A *belief state* is a set of states, informally representing the set of states that an agent considers possible. An *AGM belief revision operator* is a function $*$ that maps a belief state K and a formula ϕ to a new belief state $K * \phi$, while satisfying a particular set of postulates (Alchourrón et al., 1985).

Intuitively, the belief state $K * \phi$ incorporates the new information ϕ while keeping as much of K as consistently possible. Hence, if ϕ is consistent with

Hunter, A. and Boyarinov, K.
BRL: A Toolkit for Learning How an Agent Performs Belief Revision.
DOI: 10.5220/0010899100003116
In *Proceedings of the 14th International Conference on Agents and Artificial Intelligence (ICAART 2022) - Volume 3*, pages 753-756
ISBN: 978-989-758-547-0; ISSN: 2184-433X

K, then $K * \phi$ is just the intersection of K and ϕ. This is called *belief expansion*. But when ϕ is not consistent with K, then the set of states believed following revision will be the states where ϕ is true that are 'as close as possible' to states in K. There is a well known semantics based on orderings over possible states; we refer the reader to (Katsuno and Mendelzon, 1992) for a complete discussion.

For our purposes, the important thing to note is that AGM revision operators are only appropriate for single-shot belief revision. The problem is that the revision process requires an ordering over states, but the output is just a set of states. Hence, we do not any ordering to use for subsequent revisions. To addres *iterated belief revision*, we could move to the well-known Darwiche-Pearl approach (Darwiche and Pearl, 1997). However, this introduces many complications.

2.2 Machine Learning

In this paper, we are concerned with two kinds of Machine Learning. At the outset, we will consider *Reinforcement Learning*. Basically reinforcement learning is based on the idea that an agent is rewarded for achieving a goal. Agents learn to maximize their expected reward by choosing actions that are likely to get them closer to their goal. The well-known Q-Learning approach is a representative example of reinforcement learning; the simple, classic version of the algorithm is described in (Mitchell, 1997)

We also consider learning approaches based on classification. A *classification learning* problem involves predicting how different data points will be classified. The standard example is the *play tennis* problem, where we have information about whether an agent played tennis under different weather conditions. Using a classification learning algorithm, we can predict whether they will play tennis or not based on the past data. We assume that the reader is familiar with classification learning, as described in (Raschka and Mirjalili, 2019).

3 EXACT MATCHING

3.1 Modelling

The BRL software to be introduced in this paper is actually a collection of tools for reverse engineering a belief revision operator. In the simplest case, we assume that we have a history that consists of a set of triples (K, ϕ, K') where K' was the result when K was revised by ϕ in the past. Informally, each triple is understood to mean that there was a past situation where the agent initially believed K, then received the information ϕ and then believed K'.

We remark that this situation is not particularly reasonable in practice. The problem is that we are unlikely to know the *complete* belief state of an agent before and after an observation. However, if we do have this kind of data, there is a natural approach. We can simply check every available revision operator to see if it is consistent with the data. This is obviously a very computationally expensive approach.

3.2 Implementation

The first application of BRL we consider is the *exact matching* problem. In this case, we are looking for the set of revision operators $*$ that satisfy $K * \phi = K'$ for every historical example. This is addressed through a command line application written in C++ that essentially performs an exhaustive search.

In the *exact matching* module of BRL, the input is a comma separated text file, where each row contains an initial belief state K, a formula ϕ that was given for revision, and the new belief state that resulted. We represent states by expressions of the form $\{p:1\ \ q:0\}$, indicating that p is true and q is false. Belief states are just sets of states. For example, a single line in the history might read as follows:

```
{p:1 q:1}{p:0 q:0}, (p|q)&-(p&q), {p:1 q:0}
```

This line indicates that, when the belief state $\{\{p,q\},\emptyset\}$ was revised by $(p \vee q) \wedge \neg(p \wedge q)$, then the new belief state was $\{\{p\}\}$.

Given a set of instances in this format, BRL iterates over all possible revision operators and returns those that are consistent with the past revisions. For our purposes, the set of possible revision operators is the set of *parametrised difference(PD)* operators defined in (Peppas and Williams, 2018). We focus on PD operators because they are simple to specify, but they can capture a large class of revision operators. The output returns a set of ranks over propositional variables, which eah define a PD operator. A sample run of the program is shown in Figure 1.

The output gives a compact representation of all PD operators consistent with the input. These operators are found relatively quickly, because BRL includes an efficient revision solver and it uses OpenMP to test many operators in parallel. As a result, the run times are acceptable for small examples. For larger examples, the speed could be improved by using an ALLSAT solver for the computationally hard parts of the revision (Hunter and Agapeyev, 2020). We leave this optimization for future work.

We summarise the average run times for up to 9 variables:

```
[stanboya@OMICRON summer-2020-revision-research]$ ./learn-lr-from-dataset.out -ds ML_Datasets/generatedLRDataSet.csv

5 possible LanguageRanking(s) found:
p:0 q:1 r:0
p:0 q:2 r:1
p:1 q:1 r:0
p:2 q:1 r:0
p:1 q:2 r:0

Time Taken: 29ms

Exiting.
```

Figure 1: Exact Matching Interface.

Variables	Runtime	Variables	Run time
2	2 ms	6	730 ms
3	34 ms	7	10000 ms
4	46 ms	8	182000 ms
5	105 ms	9	30min

4 LEARNING

4.1 Modelling

Now suppose that we do not have an exact history. Instead, we have a fixed formula ϕ for revision and a fixed goal formula G. The history is a list of past initial belief states K, along with a 0-1 value that indicates if G was believed in the past when K was revised by ϕ. We refer to these as *classification histories*. Our goal in this case is to use machine learning to determine, for any possible intitial beliefs, if we should expect the agent to believe G after revision by ϕ.

We briefly argue why we feel that classification learning provides a better model for learning in this case than reinforcement learning. The fundamental idea underlying machine learning is to maximize the expected discounted reward. This means that we are assuming an agent makes choices that iteratively work towards a goal. Consider, for example, an agent that solves a maze. If we use reinforcement learning to solve this problem, the agent will learn an action policy. However, the action policy will generally be *monotonic*, in the sense that choosing a "good" move will take the agent closer to the reward.

By contrast, when we are modelling belief change with AGM revision operators, there is no such constraint. After each revision, we have no information about the ordering to use for the next revision. This means data about agent behaviour need not be monotonic, and in fact we will not have any meaningful notion of a *sequence* of revisions. For this reason, we focus on the use of classification learning algorithms for our toolkit.

4.2 Implementation

The classification learning module of BRL uses the Flask framework to implement a variety of classification algorithms. Figure 2 provides an image of the interface.

In the classification module, the available history is concerned with a single *goal formula G*. The input for BRL in this case is a *classification history*. Each instance in such a history consists of an initial belief state K, a formula ϕ, and a 0-1 value for G. A 1 value means that G was believed in the past when K was revised by ϕ. A 0 value means it was not believed. Hence, each instance has the following form:

$$\texttt{\{p:0 q:0 r:1\}\{p:1 q:1 r:0\},p|(q\&r),}0$$

Given a set of instances of this form, BRL uses classification learning algorithms to predict if $K * \phi \models G$ for new initial belief states.

For testing the system, we specify a revision operator and then generate a large set of instances based on this operator. We then use the software to determine if we should predict $K * \phi \models G$ for any new belief state K. Table 1 shows the experimental results obtained when the software was tested for 5 different algorithms on a randomly generated set of roughly 28,000 instances. The different metrics for accuracy are defined in (Raschka and Mirjalili, 2019), and they each have a maximum value of 1.

We can see that all of the ML algorithms tested actually provide reasonably accurate predictions. The accuracy of the predictions means that BRL is able to effectively determine whether or not an agent will believe the target formula, given arbitrary initial belief states. Hence, our results suggest that ML algorithms provide an effective method for learning how an unknown belief revision operator will respond to a new observation.

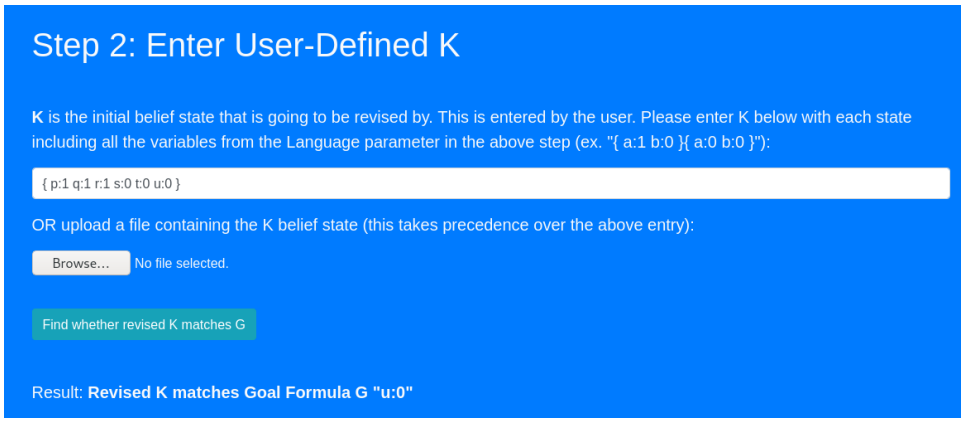

Figure 2: Classification Learning Interface.

Table 1: System Performance of Classification Learning.

ML Algorithm	Precision	Recall	F1	Running Time
BaggingClassifier (SGD)	0.91469	0.965	0.93917	214.52
RandomForestClassifier	0.96689	0.73	0.83191	4.4737
SGDClassifier	0.85027	0.795	0.82171	0.35567
BaggingClassifier (DecisionTree)	0.96970	0.96	0.96482	110.54
DecisionTreeClassifier	0.96059	0.975	0.96774	0.48756

5 CONCLUSION

The problem of deriving plausibility information for revision from examples has been addressed in theoretical work (Liberatore, 2015). However, there has been little research on the development of practical tools for learning revision operators from data. This idea is discussed in (Hunter, 2018), but that work focuses only on the feasibility of ID3 learning without any experimentation or software development. As such, BRL is really the first fully implemented sofware tool for learning revision operators from data.

Our results demonstrate that exact matching is only feasible for small examples. However, the ML approach provides a viable method to learn *something* about the revision operator an agent is using. In many practical situations, this is sufficient. More importantly, we argue that our approach here is a useful for Artificial General Intelligence in that it demonstrates how machine learning can be used together with formal methods to develop a practical reasoning tool.

REFERENCES

Alchourrón, C., Gärdenfors, P., and Makinson, D. (1985). On the logic of theory change: Partial meet functions for contraction and revision. *Journal of Symbolic Logic*, 50(2):510–530.

Darwiche, A. and Pearl, J. (1997). On the logic of iterated belief revision. *Artificial Intelligence*, 89(1-2):1–29.

Hunter, A. (2018). Learning belief revision operators. In *Proceedings of the Canadian Conference on Artificial Intelligence*, pages 239–245.

Hunter, A. and Agapeyev, J. (2020). GenC: A fast tool for applications involving belief revision. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 5219–5221.

Katsuno, H. and Mendelzon, A. (1992). Propositional knowledge base revision and minimal change. *Artificial Intelligence*, 52(2):263–294.

Liberatore, P. (2015). Revision by history. *Journal of Artificial Intelligence Research*, 52:287– 329.

Mitchell, T. M. (1997). *Machine learning, International Edition*. McGraw-Hill Series in Computer Science. McGraw-Hill.

Peppas, P. and Williams, M. (2018). Parametrised difference revision. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR)*, pages 277–286.

Raschka, S. and Mirjalili, V. (2019). *Python Machine Learning: Machine Learning and Deep Learning with Python, Scikit-Learn, and TensorFlow 2, 3rd Edition*. Expert insight. Packt Publishing, Limited.