Learning by Arguing in Argument-Based Machine Learning Framework

Matej Guid, Martin Možina, Matevž Pavlič, Klemen Turšič

Faculty of Computer and Information Science, University of Ljubljana, Slovenia

Abstract. We propose an approach for the development of argument-based intelligent tutoring systems in which a domain that can be successfully addressed by supervised machine learning is taught in an interactive learning environment. The system is able to automatically select relevant examples and counter-examples to be explained by the students. The students learn by explaining specific examples, and the system provides automated feedback on students' arguments, including generating hints. The role of an argument-based intelligent tutoring system is then to train the students to find the most relevant arguments. The students learn about the high-level domain concepts and then use them to argue about automatically selected examples. We demonstrate our approach in an online application that allows students to learn through arguments with the goal of improving their understanding of financial statements.

Keywords: intelligent tutoring systems, learning by arguing, argument-based machine learning, automated feedback generation, financial statements

1 Introduction

When students collaborate in argumentation in the classroom, they are arguing to learn. Argumentation can help learners to accomplish a wide variety of important learning goals. [2] It involves elaboration, reasoning, and reflection. These activities have been shown to contribute to deeper conceptual learning [3]. Effective learning by arguing involves making knowledge explicit: learners that provide explanations, or make explicit the reasoning underlying their problem solving behavior, show the most learning benefits [4].

It is often the case that domains of interest can be represented by numerous learning examples. These learning examples can typically be described by various features and may contain labels such as 'good', 'bad' or 'ugly'. A domain described by labeled learning examples can be tackled by *supervised machine learning* algorithms. Supervised machine learning technique in which the algorithm attempts to label each learning example by choosing between two or more different classes is called *classification*. For instance, a classification task would be to learn a prediction model that distinguishes between successful and less successful companies based on their financial statements.

We propose an approach to the development of argument-based intelligent tutoring systems using argument-based machine learning (ABML) [12] framework. In principle, it allows any domain that can be successfully addressed by supervised machine learning

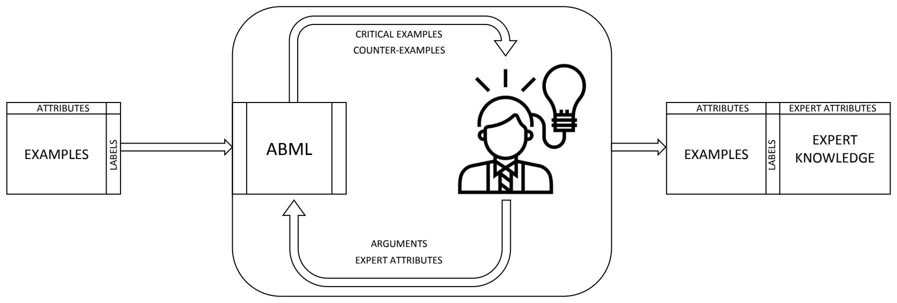


Fig. 1. ABML Knowledge Refinement Loop with the domain expert.

to be taught in an interactive learning environment. It has been shown that ABML provides the opportunity to develop interactive teaching tools that are able to automatically select relevant examples and counter-examples to be explained by the student [15]. We extend that approach with automated feedback on students' arguments, including the generation of hints. In addition, we have developed a web application that can be easily adapted to various learning domains. It allows students to learn by arguing with a very clear objective: to improve their understanding of particular learning domains.

The chosen experimental domain was financial statement analysis. More concretely, estimating credit scores or the creditworthiness of companies. Our aim was to obtain a successful classification model for predicting the credit scores and to enable the students to learn about this rather difficult subject.

In the experiments, both the teacher and the students were involved in the interactive process of knowledge elicitation based on the ABML paradigm, receiving the feedback on their arguments. The aim of the learning session with the teacher was in particular to obtain advanced concepts (features) that describe the domain well, are suitable for teaching and also enable successful predictions. This was done with the help of a financial expert. In the tutoring sessions, the students learned about the intricacies of the domain and looked for the best possible explanations for automatically selected examples, using the teacher's advanced concepts in their arguments. The system is also able to track the student's progress in relation to these selected concepts.

2 Argument-Based Machine Learning

Argument-based machine learning (ABML) [12] is machine learning, extended by concepts from argumentation. In ABML, arguments are typically used as a means for users (e.g. domain experts, students) to elicit some of their knowledge by explaining the learning examples. The users only need to concentrate on one specific case at a time and impart knowledge that seems relevant for this case. They provide the knowledge in the form of arguments for the learning examples and not in the form of general domain knowledge.

ABML Knowledge Refinement Loop [11] enables an interaction between a machine learning algorithm and a domain expert (see Fig. 1). It is a powerful knowledge acquisition tool capable of acquiring expert knowledge in difficult domains [7–9, 13]. The loop allows the expert to focus on the most critical parts of the current knowledge base and helps him to discuss automatically selected relevant examples. The expert only needs to explain a single example at the time, which facilitates the articulation of arguments. It also helps the expert to improve the explanations through appropriate counter-examples.

We use the ABCN2 [12] method, an argument-based extension of the well-known CN2 method [5], which learns a set of unordered probabilistic rules from examples with attached arguments, also called *argumented examples*.

2.1 ABML Knowledge Refinement Loop

In this section, we give a brief overview of the steps in the ABML knowledge refinement loop from the perspective of the expert:

Step 1: Learn a hypothesis with ABCN2 using the given data.
Step 2: Find the “most critical” example and present it to the student. If a critical example cannot be found, stop the procedure.
Step 3: Expert explains the example; the explanation is encoded in arguments and attached to the critical example.
Step 4: Return to step 1.

In the sequel, we explain (1) how to select critical examples and (2) how to obtain all necessary information for the selected example.

Identifying Critical Examples The arguments given to the critical examples cause ABCN2 to learn new rules that cover these examples. A critical example is an example with a high probabilistic prediction error. The probabilistic error can be measured in different ways. We use the Brier Score with a k-fold cross-validation repeated n times (e.g. $n = 4, k = 10$), so that each example is tested n times. The most problematic example is therefore the one with the highest average probabilistic error over several repetitions of the cross-validation procedure.

Improving a Expert’s Arguments In the third step of the above algorithm, the expert is asked to explain the critical example. With the help of the expert’s arguments, ABML will sometimes be able to explain the critical example, while sometimes this is still not entirely possible. Then we need additional information from the expert where the counter-examples come into play. The following five steps describe this idea:

Step 3a: Explain the critical example. The expert is asked the following question: “Why is this example in the class as given?” The answer can be either “I don’t know” (the expert cannot explain the example) or the expert can specify an argument that confirms the class value of the example. If the system receives the answer “don’t know”, it stops the process and tries to find another critical example.
Step 3b: Add arguments. The argument is usually given in natural language and must be translated into domain description language (attributes). One argument supports its allegation with a number of reasons. The role of the expert is also to introduce new concepts to the domain. These concepts are added as new attributes so that they can appear in an argument attached to the example.

Step 3c: Discover counter-examples. A *counter-example* is an example from the opposite class that is consistent with the expert's argument.

Step 3d: Improve arguments. The expert must revise the first argument in relation to the counter-example. This step is similar to steps 1 and 2 with one essential difference; the expert is now asked: "Why is the critical example in one class and why the counter-example in the other?" Note that the argument is always attached to the critical example (and never to the counter-example).

Step 3e: Return to step 3c when a counter-example is found.

In the context of argument-based intelligent tutoring systems, the ABML Knowledge Refinement Loop is used to obtain expert concepts in the form of attributes that describe the domain well, are suitable for teaching and also enable successful rule-based models that achieve high predictive accuracy in the given domain.

3 Arguing to Learn

The ABML Knowledge Refinement Loop can also be used to create meaningful learning experience for a student (see Fig. 2). To achieve this, the intelligent tutoring system must be able to do more than just automatically select relevant examples and counter-examples to be explained by the students. Namely, there should also be automated feedback on the students' arguments, including the generation of hints. The role of an argument-based intelligent tutoring system is then to train students to find the most relevant arguments. Students are encouraged to formulate their arguments using concepts presented by the expert. The students get to know the high-level domain concepts and then use them to argue about automatically selected examples.

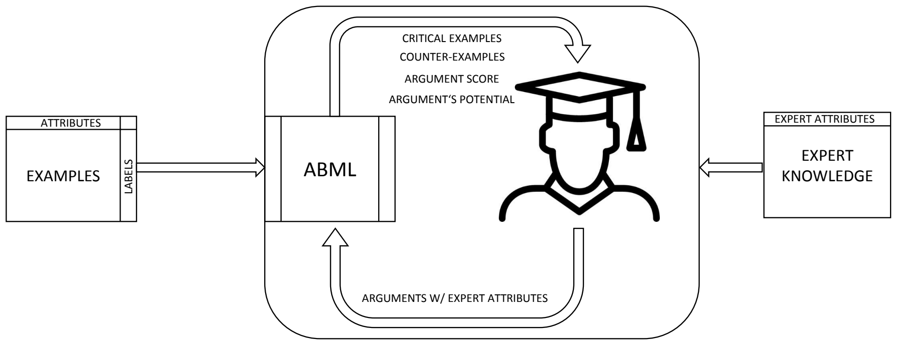


Fig. 2. ABML Knowledge Refinement Loop with the student.

We describe three types of feedback on students' arguments, all of which are automatically generated by the underlying machine learning algorithm to help the students construct better arguments and therefore learn faster.

3.1 Counter-Examples

The feedback comes in three forms. The first are the counter-examples that are already inherent in the ABML process. A counter-example is an instance from the data that is consistent with the reasons in the given argument, but whose class value is different from the conclusion of the argument. Therefore, the counter-example is a direct rebuttal to the student's argument. The student must either revise the original argument or accept the counter-example as an exception.

In contrast to earlier applications of the ABML Knowledge Refinement Loop (e.g. [15]), our implementation allows the simultaneous comparison of the critical example with several counter-examples. We believe that this approach allows the student to argue better, as some of the counter-examples are less relevant than others.

3.2 Assessment of the Quality of the Argument

The second type of feedback is an assessment of the *quality* of the argument. A good argument gives reasons for decisions that distinguish the critical example from examples from another class. A possible formula for estimating the quality could therefore be to simply count the number of counter-examples: An argument without counter arguments is generally considered to be a strong argument. However, this method considers very specific arguments (e.g. arguments that only apply to the critical example) to be good. Such specific knowledge is rarely required, we usually prefer general knowledge, which can be applied in several cases.

Therefore, we propose to use the *m-estimate* of probability to estimate the quality of an argument. The formula of the m-estimate balances between the prior probability and the probability assessed from the data:

$$Q(a) = \frac{p + m \cdot p_a}{p + n + m}. \tag{1}$$

Here, p is the number of all covered instances that have the same class value as the critical example, and n is the number of all data instances of another class covered by the argument. We say that an argument covers an instance if the reasons of the argument are consistent with the feature values of the instance. The prior probability p_a and the value m are the parameters of the method used to control how general arguments should be. We estimated the prior probability p_a from the data and set m to 2.

Consider, for example, the argument given to the following critical example:

CREDIT SCORE is *good* because EQUITY RATIO is *high*.

The student stated that this company has a good credit score, as its equity ratio (the proportion of equity in the company's assets) is high. Before the method can evaluate such an argument, it must first determine the threshold value for the label "high". With the entropy-based discretization method, the best threshold for our data was about 40, hence the grounded argument is:

CREDIT SCORE is *good* because EQUITY RATIO > 40 (51, 14).

The values 51 and 14 in brackets correspond to the values p and n, respectively. The estimated quality of this argument using the m-estimate is thus 0.77.

3.3 Potential of the Argument

The last and third type of feedback is the *potential* of the argument. After the student has received an estimate of the quality of his argument, we also give him an estimate of how much the quality would increase if he had improved the argument.

The quality of an argument can be improved either by removing some of the reasons or by adding new reasons. In the first case, we search the existing reasons and evaluate the argument at each step without this reason. For the latter option, we attach the student's argument to the critical example in the data and use the ABCN2 algorithm to induce a set of rules consistent with that argument (this is the same as Steps 3 and 1 in the knowledge refinement loop). The highest estimated quality (of pruned and induced rules) is the potential of the argument provided.

For example, suppose the student has improved his previous argument by adding a new reason:

CREDIT SCORE is *good* because EQUITY RATIO is *high* and CURRENT RATIO is *high*.

The quality of this argument is 0.84. With the ABML method we can induce several classification rules containing EQUITY RATIO and CURRENT RATIO in their condition parts. The most accurate one was:

if NET INCOME $>$ €122,640 and EQUITY RATIO $>$ 30 and CURRENT RATIO $>$ 0.85 then CREDIT SCORE is *high*.

The classification accuracy (estimated with m-estimate, same parameters as above) of the rule is 0.98. This is also the potential of the above argument, since the quality of the best pruned argument is lower (0.77). The potential tells the student that his argument can be improved from 0.84 to 0.98.

4 Case Study

4.1 Domain Description

Credit risk assessment plays an important role in ensuring the financial health of financial and non-financial institutions. Based on a *credit score*, the lender determines whether the company is suitable for lending and how high the price should be. The credit scores are assigned to companies on the basis of their annual financial statements. Arguing what constitutes the credit score of a particular company can significantly improve the understanding of the financial statements [6].

For the machine learning problem, we distinguished between companies with good credit scores and those with bad credit scores. We obtained annual financial statements and credit scores for 325 Slovenian companies from an institution specialising in issuing credit scores. The annual financial statements show the company's business activities in the previous year and are calculated once a year. There were 180 examples of companies with a *good* score and 145 companies with a *bad* score.

At the beginning of the machine learning process, the domain expert selected 25 features (attributes) describing each company. Of these, 9 were from the Income Statement (net sales, cost of goods and services, cost of labor, depreciation, financial expenses, interest, EBIT, EBITDA, net income), 11 from the Balance Sheet (assets, equity, debt, cash, long-term assets, short-term assets, total operating liabilities, short-term operating liabilities, long-term liabilities, short-term liabilities, inventories), 2 from the Cash Flow Statement (FFO - fund from operations, OCF - operating cash flow), and the remaining 3 were general descriptive attributes (activity, size, ownership type).

4.2 Knowledge Elicitation from the Financial Expert

The goal of the knowledge elicitation from the expert is (1) to obtain a rule-based model consistent with his knowledge, and (2) to obtain relevant description language in the form of new features that would describe the domain well and are suitable for teaching. In the present case study, the knowledge elicitation process consisted of 10 iterations. The financial expert introduced 9 new attributes during the process. The new attributes also contributed to a more successful rule model: in the interactive sessions with students (see Section 4.3), using the expert's attributes in arguments lead to classification accuracies up to 97%.

The target concepts obtained in the knowledge elicitation process were: Debt to Total Assets Ratio, Current Ratio, Long-Term Sales Growth Rate, Short-Term Sales Growth Rate, EBIT Margin Change, Net Debt To EBITDA Ratio, Equity Ratio, TIE - Times Interest Earned, ROA - Return on Assets. An interested reader can find descriptions of these concepts in introductory books to financial accounting such as [10].

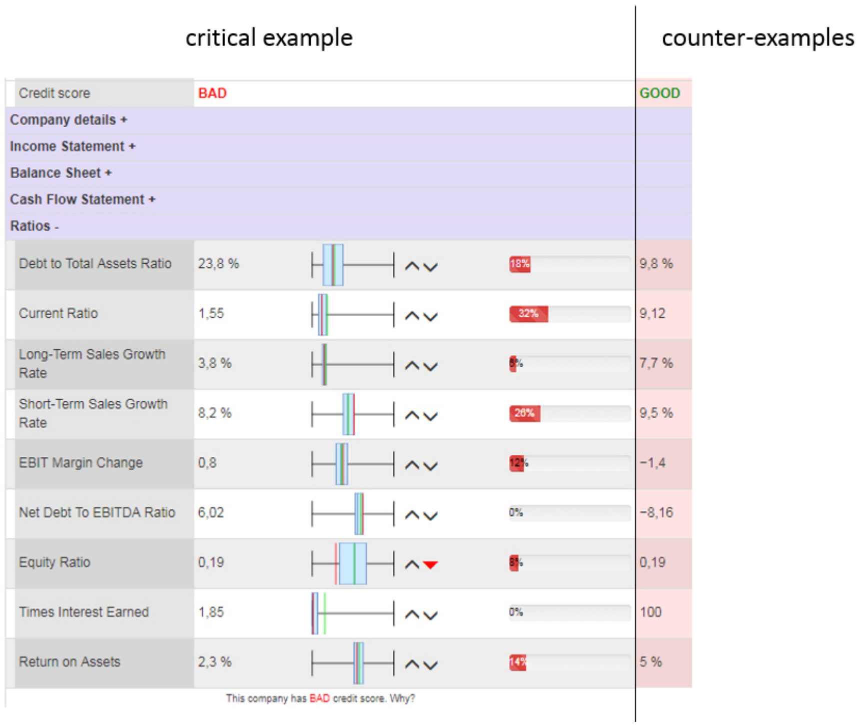
4.3 Interactive Learning Session

In the learning session, each student looks at the annual financial statements of automatically selected companies and faces the challenge of arguing whether a particular company has a good or bad credit score. Students are instructed to use in their arguments the expert features obtained through the process of knowledge elicitation described in Section 4.2. This means that the goal of the interaction is that the student is able to explain the creditworthiness of a company using the expressive language of the expert. We will now describe an iteration of one of the learning sessions.

The financial statement and the value of the expert attributes of the training example B.132 were shown to the student (see left part of Fig. 3). The student's task was to explain why this company has a bad credit score.

The student's explanation was: "The company has a certain profit (looking at EBIT and EBITDA in the financial statement) and sales are growing, but also has significant liabilities. Only a small part of the total assets is financed by stockholders, as opposed to creditors. The main reason for the bad credit score seems to be low Equity Ratio." The student also mentioned a high Net Debt To EBITDA Ratio value – note that this attribute introduced by the financial expert is one of the target concepts that the student has to master – but was nevertheless satisfied with the following argument:

CREDIT SCORE is *bad* because EQUITY RATIO is *low*



Credit score	BAD				GOOD
Company details +					
Income Statement +					
Balance Sheet +					
Cash Flow Statement +					
Ratios -					
Debt to Total Assets Ratio	23,8 %		^ v	18%	9,8 %
Current Ratio	1,55		^ v	32%	9,12
Long-Term Sales Growth Rate	3,8 %		^ v	6%	7,7 %
Short-Term Sales Growth Rate	8,2 %		^ v	26%	9,5 %
EBIT Margin Change	0,8		^ v	12%	−1,4
Net Debt To EBITDA Ratio	6,02		^ v	0%	−8,16
Equity Ratio	0,19		^ ▼	8%	0,19
Times Interest Earned	1,85		^ v	0%	100
Return on Assets	2,3 %		^ v	14%	5 %

This company has BAD credit score. Why?

Fig. 3. The student was asked to improve the argument. The attribute values of the critical example are shown on the left and the values of a counter-example are on the right.

New rules were obtained and the system identified several counter-examples, such as the one shown in the right side of Fig. 3. The estimated quality of the student's argument was 0.89, which means that the student actually gave a rather good reason why the credit score of the critical example is bad, but, as implied by the potential, can still be significantly improved. The counter-example shown in Fig. 3 also has a low Equity Ratio, but comes from the opposite class. That is, it has a good credit score *despite* low Equity Ratio.

The user noticed a very big difference in the values of Times Interest Earned (TIE). He looked up the definition (available in the application) and found that this attribute indicates how many times a company's earnings cover its interest payments and indicates the probability that a company is (not) able to meet its interest payment obligations. If the student had clicked on "Hints" button in the application, he would get TIE among the possible options to improve the argument. However, he would still have to find out for himself whether a high or low value of TIE indicates good or bad credit score. The student decided to extend the argument:

CREDIT SCORE is *bad* because EQUITY RATIO is *low* and TIE is *low*

This time the potential of the argument implied that the argument could not be further improved, therefore the student demanded a new critical example.

5 Assessment

The main objective of the experiments presented in this section was to observe whether automatically generated feedback can enable meaningful interaction suitable for learning. The authors in [1] point out that most studies associated with arguing to learn report great individual differences in the use of tools. They suggest that care must be taken not to fall into the trap of deciding too early which use is correct and which one is not. Since there are many possible solutions and applications of our learning approach through argumentation with the argument-based machine learning framework, we see the results of the pilot experiments described in the sequel of this chapter more as a proof of concept than as a precise evaluation of the concrete applications.

We conducted a pilot experiment with three students. It consisted of 31 iterations such as the one described in Section 4.3. All students started the interactive learning session with the same data. The average time per session was 2.83 hours. The average number of arguments analyzed was 2.49 (s = 0.37) per iteration. The student typically analyzed more than one argument per iteration, since, with the help of the automatically generated feedback, the arguments were refined several times.

To assess the students' learning performance, we asked them in a post-test to assign credit scores to 30 previously unseen examples. The students' classification accuracy was 87%. We see this as a very positive result, considering that these students had a rather poor understanding of financial statements only a couple of hours earlier and were not aware of the high-level concepts reflected in the expert attributes.

In another experiment, nine students took an online pre-test in which they were asked to tell the credit score of five companies, provide arguments for their decisions and express their confidence when making the decision on a scale from 1 (low) to 5 (high). Then they used the online tool (see Fig. 3) for about 45 minutes. In a subsequent online post-test, the students were asked exactly the same questions as in the pre-test. We found that the estimated quality of the students' arguments increased on average from 0.85 to 0.87, while the confidence increased on average from 2.70 to 3.58. It appeared that at the end of the process the students could confidently use the high-level concepts introduced by the financial expert in their arguments.

6 Conclusions

We introduced the core mechanisms behind a novel kind of argument-based intelligent tutoring systems based on argument-based machine learning. The main mechanism enabling an argument-based interactive learning session between the student and the computer is called *argument-based machine learning knowledge refinement loop*. By using a machine learning algorithm capable of taking into account a student's arguments, the system automatically selects relevant examples and counter-examples to be explained by the student. The role of an argument-based intelligent tutoring system is to train students to find the most relevant arguments. The students get to know the high-level domain concepts and use them to argue about automatically selected examples. In the interactive learning session, they receive three types of automatically generated feedback: (1) a set of counter-examples, (2) a numerical evaluation of the quality of the argument, and (3) the potential of the argument or how to extend the argument to make it more effective.

The beauty of this approach to developing intelligent tutoring systems is that, at least in principle, any domain that can be successfully tackled by supervised machine learning can be taught in an interactive learning environment that is able to automatically select relevant examples and counter-examples to be explained by the students. To this end, as a line of future work, we are considering the implementation of a multi-domain online learning platform based on argument-based machine learning, taking into account the design principles of successful intelligent tutoring systems [14].

References

1. Andriessen, J., Baker, M.: Arguing to learn. In: Sawyer, R.K. (ed.) The Cambridge Handbook of the Learning Sciences, chap. 22, pp. 439–460. Cambridge Handbooks in Psychology, Cambridge University Press (2014)
2. Andriessen, J., Baker, M., Suthers, D.D.: Arguing to learn: Confronting cognitions in computer-supported collaborative learning environments, vol. 1. Springer Science & Business Media (2013)
3. Bransford, J.D., Brown, A., Cocking, R.: How people learn: Mind, brain, experience, and school. Washington, DC: National Research Council (1999)
4. Chi, M.T., VanLehn, K.A.: The content of physics self-explanations. The Journal of the Learning Sciences 1(1), 69–105 (1991)
5. Clark, P., Boswell, R.: Rule induction with CN2: Some recent improvements. In: European Working Session on Learning. pp. 151–163. Springer (1991)
6. Ganguin, B., Bilardello, J.: Standard and Poor's Fundamentals of Corporate Credit Analysis. McGraw-Hill Professional Publishing (2004)
7. Groznik, V., Guid, M., Sadikov, A., Možina, M., Georgiev, D., Kragelj, V., Ribarič, S., Pirtošek, Z., Bratko, I.: Elicitation of neurological knowledge with argument-based machine learning. Artificial Intelligence in Medicine 57(2), 133–144 (2013)
8. Guid, M., Možina, M., Groznik, V., Georgiev, D., Sadikov, A., Pirtošek, Z., Bratko, I.: ABML knowledge refinement loop: A case study. In: Foundations of Intelligent Systems. pp. 41–50. Springer Berlin Heidelberg (2012)
9. Guid, M., Možina, M., Krivec, J., Sadikov, A., Bratko, I.: Learning positional features for annotating chess games: A case study. In: Lecture Notes in Computer Science. vol. 5131, pp. 192–204 (2008)
10. Holt, R.: Financial Accounting: A Management Perspective. Ivy Learning Systems (2001)
11. Možina, M., Guid, M., Krivec, J., Sadikov, A., Bratko, I.: Fighting knowledge acquisition bottleneck with argument based machine learning. In: The 18th European Conference on Artificial Intelligence (ECAI). pp. 234–238. Patras, Greece (2008)
12. Možina, M., Žabkar, J., Bratko, I.: Argument based machine learning. Artificial Intelligence 171(10/15), 922–937 (2007)
13. Možina, M., Lazar, T., Bratko, I.: Identifying typical approaches and errors in prolog programming with argument-based machine learning. Expert Systems with Applications (2018)
14. Woolf, B.P.: Building Intelligent Interactive Tutors: Student-centered strategies for revolutionizing e-learning. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2008)
15. Zapušek, M., Možina, M., Bratko, I., Rugelj, J., Guid, M.: Designing an interactive teaching tool with ABML knowledge refinement loop. In: International Conference on Intelligent Tutoring Systems. pp. 575–582. Springer (2014)