Learning Feedback in Intelligent Tutoring Systems

Sebastian Gross Bassam Mokbel Barbara Hammer
Niels Pinkwart

(This is a preprint of the publication [9], as provided by the authors.)

Abstract

Intelligent Tutoring Systems (ITSs) are adaptive learning systems that aim to support learners by providing one-on-one individualized instruction. Typically, instructing learners in ITSs is build on formalized domain knowledge and, thus, the applicability is restricted to well-defined domains where knowledge about the domain being taught can be explicitly modeled. For ill-defined domains, human tutors still by far outperform the performance of ITSs, or the latter are not applicable at all. As part of the DFG priority programme "Autonomous Learning", the FIT project has been conducted over a period of 3 years pursuing the goal to develop novel ITS methods, that are also applicable for ill-defined problems, based on implicit domain knowledge extracted from educational data sets. Here, machine learning techniques have been used to autonomously infer structures from given learning data (e.g., student solutions) and, based on these structures, to develop strategies for instructing learners.

1 Introduction

Intelligent Tutoring Systems (ITSs) have made great strides in recent years. The goal of these educational technology systems is to provide intelligent, one-on-one, computer-based support to students as they learn to solve problems in a type of instruction that is often not available because of scarce (human) resources [21]. The basic rationale underlying traditional ITS systems is that by using formalized domain knowledge and dynamic student models, comparing student's problem solving steps to the explicit domain models, it is possible to adapt instruction to the needs of specific students and effectively support their learning. Two prominent approaches that both use such explicit domain models have demonstrated impressive practical success in teaching domains such as algebra, geometry, programming, database design or UML [13, 14]:

Constraint-Based Tutors [20] rely on checking student solutions for the fulfillment of certain correctness constraints (and give feedback, should these be violated), Cognitive Tutors [2] are based on a detailed and formalized cognitive model that represents the process of building correct solutions.

While leading to impressing results, e.g., in mathematics education in U.S. High Schools, ITS research still faces two problems: (i) current ITS systems have their advantages only in well-structured and comparably narrow domains since they need to judge whether and why a given student answer is correct or wrong; (ii) to achieve this goal, current ITS systems require an exact formalization of the underlying domain knowledge which is usually a substantial amount of work: researchers have reported on 100-1000 hours of authoring time needed for one hour of instruction [19].

In order to extend the applicability of ITSs to more ill-structured problems and to reduce the effort for building such systems, the overarching goal of the FIT project has been to research novel approaches that do not require explicit domain models but infer domain knowledge autonomously from educational data sets.

2 Challenges & Objectives

If ITSs cannot rely on explicit models or complete domain knowledge, it may be possible to acquire information about the domain in terms of educational data sets consisting of examples (e.g., solution attempts made by students or sample solutions created by experts) how to solve a given problem. Here, machine learning constitutes a key technology to infer meaningful information from given examples. Due to the nature of ill-defined domains and the inherent noise and errors in typical application data (i.e. student solution attempts), statistical machine learning techniques in unsupervised or semi-supervised settings such as data clustering seem appropriate. There are two challenges for machine learning methods that aim at structuring educational data sets:

1. Typical data in ITSs is not given by Euclidean vectors as is common in statistical machine learning. Rather, structural elements play an important role.

2. Models inferred by machine learning should constitute an interface towards human understandable feedback mechanisms. This rules out typical black box techniques which could be used to learn a function (such as whether a student solution is correct) but which do not explain why they take a decision.

To account for model interpretability, we decided to focus on intuitive prototype-based techniques, which represent the data in terms of typical examples, and which base their result on a comparison of the given data to the learned prototypes. Considering both challenges, we identified four objectives. First, prototype-based machine learning techniques need to be enhanced for tree structured data (e.g., XML documents) instead of

simple vectors such that flexible and interpretable statistical machine learning becomes possible in the context of complex structural representations. Second, machine learning methods need to be devised to autonomously adapt (alignment) metrics for data comparison simultaneously to the inference of data clusters. The metric has to take into account structural constituents such as basic texts and their relation as represented, e.g., by XML documents, possibly incorporating problem dependent invariances. An expected result of the first two objectives is the ability to automatically cluster (sample) solutions of ill-defined problems, resulting in sets of similar and good solutions (with possibly different solution pathways and strategies) and sets of similar but incorrect attempts (with different errors made, and different strategies pursued by the students). Therefore, third, it needs to be determined how, based on this material, learners can be instructed (e.g., by providing relevant and helpful feedback to learners in case of mistakes) under the condition of uncertainty and without explicit domain models. The main goal of any Intelligent Tutoring System is to deliver efficient and effective training to students. Consequently, as the fourth objective, machine learning methods and ITS methods need to be tested in empirical studies for their strengths and weaknesses. Here, the measurements we are interested in include both tutoring effectiveness and efficiency in terms of student's learning gains as well as cost-benefit studies, comparing the development costs for tutors that use this novel approach to those of other ITS systems.

3 Advances & Results

We initially preprocessed existing data sets (Java programs, UML diagrams and argumentation maps) in order to achieve a structure that is capable of representing different data types (e.g., plain text or XML based data). This resulted in a common representation: an annotated graph with nodes representing structural elements in a solution and edges representing dependencies between those nodes [15]. A subset of the data set was manually coded in terms of quality for the purpose of later comparison with automated assessments. Also, we defined the term "solution space" in more detail, including both a technical dimension on how to analyze such spaces of learner (and sample) solutions using machine learning techniques, and a pedagogical dimension on how to give useful feedback to a learner's solution within such a space [10]. In addition, we identified several evaluation methods (e.g., pre-/post-testing) and criteria (e.g., learner's action after and before next instruction) that are suitable for measuring the impact of FIT ITS methods on human learning.

To provide meaningful automated feedback via machine learning methods, the representation of input data is a key factor. We have taken a general approach, which relies on a data representation in terms of pairwise proximities only. The proximity values are provided by dissimilarity measures, which offer several benefits over a classical feature

description of the data: When processing a pair of solutions, dissimilarity measures can take into account structural aspects in the given encoding, and treat the solutions' constituents in direct comparison with each other. Dissimilarities are also not restricted to distances in a Euclidean vector space, and they are not necessarily metric. We only assume symmetry and vanishing self-dissimilarities, to ensure that a pseudo-Euclidean embedding exists, which provides a sound theoretical basis to represent the underlying data space.

Regarding objective 1, we developed a general framework which allows to transfer prototype-based machine learning methods to dissimilarities [11]. Therein, prototypes are represented implicitly by linear combinations in the underlying pseudo-Euclidean space. We have shown that these *relational* clustering and classification techniques can structure solution spaces, and yield prototypical solutions for a direct interpretation of the data-driven model [15, 10]. Regarding objective 2, we developed a domain-agnostic dissimilarity measure, based on a highly parameterized sequence alignment[1] over traversals of subgraphs. To adapt this measure to a given domain or learning task, the parameters (in this case the scoring of the alignment function) have to be adjusted accordingly. We proposed an approach for an autonomous adaptation of the scoring parameters, in conjunction with classifier training [18, 16, 17]. This facilitates model accuracy for the given solution space and also offers the possibility to interpret semantic implications of the adapted dissimilarity measure.

With respect to objective 3, we investigated how, based on structured solution spaces, learners can be instructed in problem solving processes. We identified, implemented and (partially) tested three main principles so far.

(I) Example-based learning is suitable for situations where explicit domain knowledge is not available but example solutions provide an implicit description of the solution space. It has been shown to be effective, encouraging learner's self-reflection by guiding her through the learning process using (worked) examples [22]. In FIT, **self-reflection and example based learning** can be implemented by asking learners to compare their own solution to a similar (but not identical) element of the solution space, to self-explain the observable differences and to think about possible mistakes and how to fix them.

(II) Model-tracing ITSs have shown to successfully support learning in well-defined domains. In these systems, a learner is supported via feedback provision which is calculated based on a **cognitive model** which consists of correct and buggy rules. We adapted this approach to ill-defined domains by using the solution space as a heuristic data driven substitute for the cognitive model. The underlying principle is that if a learner asks for feedback, the solution space is searched for an element that is not only similar to the learner solution, but (even more) likely a next step in a correct (or buggy) solution process. The system then uses this element for feedback provision –

[1] `http://openresearch.cit-ec.de/projects/tcs`

e.g., by contrasting it to the current student solution, or by highlighting the elements in the student solution where it differs from its counterpart.

(III) Since a solution space consists of solutions of several learners, another possible form of feedback provision for FIT ITSs is to bring learners together and stimulate **collaborative learning** interactions. For instance, peer tutoring or peer reviewing sessions can lead to a fruitful learning. The FIT solution space can be used for these approaches: it not only contains information about users, but also about the similarity of learner solutions which can be used for the assignment of peer reviews or the formation of learning groups.

To apply and test these feedback provision strategies in multiple domains and with heterogeneous tools, we designed a middleware architecture [6] that provides typical components of ITSs and prototype-based machine learning via standard interfaces. It implements the typical components used in ITSs: a student model stores information about learners and their actions; a pedagogical module contains pedagogical strategies implementing the above mentioned feedback principles.

As argued in Section 1, ITSs have shown to be effective in a variety of teaching domains. This includes several successful ITS approaches for teaching programming. The LISP Tutor, a very early cognitive tutor for programming, helps students learn the LISP programming language by providing feedback to learners based on cognitive models of ideal solutions [1]. Coull and colleagues [4] propose to parse compiler messages and to compare them to a domain model in order to help learners find mistakes in their programs. J-LATTE is a constraint-based tutor for Java programming that checks the correctness of students' programs using a model of precoded constraints [12].

In comparison to the above mentioned tutoring systems that make use of predefined (cognitive) domain models, our approach uses machine learning techniques that automatically infer (cognitive) states from a given solution space. We utilized the previously introduced middleware architecture to build a tutoring system for Java programming. This tutor implements feedback principles (I) and (II) and provides example-based feedback in order to help students reflect on their solutions, find mistakes and give explanations how to solve a given problem. In contrast to the example-based learning approach pursued by the NavEx tutor [3], our approach is based not only on complete sample solutions but also includes steps towards a complete solution (as illustrated in Fig. 1) in order to reduce learners' cognitive load and to adaptively instruct them depending on their actual state in problem-solving.

Focusing on pedagogical principles of feedback provision as described in (I) and (II) and on the utility of prototype based machine learning techniques for these principles, we conducted several pilot studies to address objective 4. In an expert survey [5], we asked human Java tutors whether example-based feedback provision methods as developed in FIT make sense. The results of this survey supported the hypothesis that the example-based feedback provision methods can be beneficial for learning.

We then conducted two Wizard-of-Oz studies [10, 7] asking experts to simulate

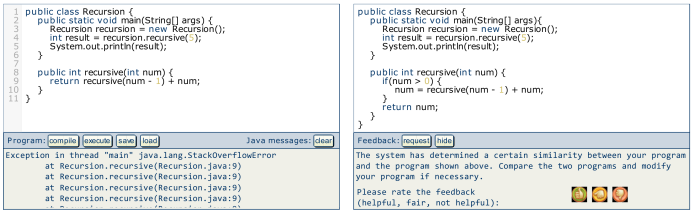

Figure 1: Application of feedback principles (I) and (II) in the domain of Java programming. A student is prompted to reflect her solution attempt (illustrated on the left) and to compare it to the next step of a correct example (illustrated on the right).

the behavior of a FIT ITS. In these studies, we examined two different example-based feedback provision strategies in the domain of Java programming: highlighting and/or contrasting similarities among learner solutions and sample solutions (as illustrated in Fig. 1). The results show a positive impact of the instruction that was positively rated by the learners [10], and the study also confirmed that manual feedback generation indeed means a lot of work, and that the ITS feedback methods have been assessed as potentially helpful in most cases [7].

Finally, we conducted a field study in an introductory programming course for Java programming, comparing the appropriateness of different exemplar selection strategies (user or sample solutions, full or partial solutions) for feedback provision. This study yielded that examples created by experts are more beneficial to support learning than using solution attempts of other learners as examples. Here, providing parts of sample solutions has shown to be more beneficial for learning than revealing the complete sample solution [8].

4 Future Research: DynaFIT

In FIT, we examined novel approaches for ITSs on how to instruct learners in problem solving processes based on implicit models of domain knowledge (represented by structured solution spaces). Several empirical evaluations have shown a positive impact on learning, however, we also found a limitation of this approach: since a FIT ITS provides feedback depending on learner's current activity in problem-solving only, it disregards past activities. This could result in a loop where a learner receives the same (inadequate) feedback again and again, without being able to proceed in problem-solving. The overarching goal of DynaFIT, a successor project to FIT, is to enhance feedback strategies in ITSs for ill-defined domains so that self-regulated learning is supported and the activity of a learner during a sequence of learning tasks is taken into account: the ITS system should autonomously react and adapt to this dynamic human

learner behavior. This objective requires the development of machine learning strategies which support an autonomous information transfer across different learning tasks by means of a common representation. The following four objectives will be central for DynaFIT:

1. Devising machine learning techniques which allow to transfer information across different learning tasks in a given domain, this way enabling an efficient representation and visualization of learners' behavior in problem-solving;
2. Devising strategies for using these representation and visualization mechanisms in order to help human learners improve their metacognitive skills (specifically, helping them decide when best to ask for feedback);
3. Developing relevance learning techniques which can predict the potential usefulness of giving feedback at a certain point in time based on the learners' behavior;
4. Investigating mechanisms for adapting the mode of feedback provision to learners specific needs considering their development over time.

The DynaFIT project started in the beginning of 2015 and will last another 3 years. The project comprises theoretical aspects regarding machine learning techniques and feedback principles as well as their technical implementation in Intelligent Tutoring Systems. It also includes empirically evaluating the effectiveness of these feedback principles. This work was supported by the German Research Foundation (DFG) under the grant "FIT - Learning Feedback in Intelligent Tutoring Systems." (PI 767/6 and HA 2719/6).

References

[1] J. R. Anderson, F. G. Conrad, and A. T. Corbett. Skill acquisition and the lisp tutor. *Cognitive Science*, 13(4):467–505, 1989.

[2] J. R. Anderson, A. T. Corbett, K. R. Koedinger, and R. Pelletier. Cognitive tutors: Lessons learned. *Journal of the Learning Sciences*, 4:167–207, 1995.

[3] P. Brusilovsky and M. Yudelson. From webex to navex: Interactive access to annotated program examples. *Proceedings of the IEEE*, 96(6):990 –999, june 2008.

[4] N. Coull, I. Duncan, J. Archibald, and G. Lund. Helping Novice Programmers Interpret Compiler Error Messages. In *Proceedings of the 4th Annual LTSN-ICS Conference*, pages 26–28. National University of Ireland, Galway, Aug. 2003.

[5] S. Gross, B. Mokbel, B. Hammer, and N. Pinkwart. Feedback provision strategies in intelligent tutoring systems based on clustered solution spaces. In J. Desel, J. M. Haake, and C. Spannagel, editors, *Tagungsband der 10. e-Learning Fachtagung Informatik (DeLFI)*, number P-207 in GI Lecture Notes in Informatics, pages 27–38, Bonn, Germany, 2012. GI.

[6] S. Gross, B. Mokbel, B. Hammer, and N. Pinkwart. Towards a domain-independent its middleware architecture. In N.-S. Chen, R. Huang, Kinshuk, Y. Li, and D. G. Sampson, editors, *Proceedings of the 13th IEEE International Conference on Advanced Learning Technologies (ICALT)*, pages 408–409, Los Alamitos, CA, 2013. IEEE Computer Society Press.

[7] S. Gross, B. Mokbel, B. Hammer, and N. Pinkwart. Towards providing feedback to students in absence of formalized domain models. In H. C. Lane, K. Yacef, J. Mostow, and P. Pavlik, editors, *Proceedings of the 16th International Conference on Artificial Intelligence in Education (AIED)*, volume 7926 of *Lecture Notes in Computer Science*, pages 644–648, Berlin, Germany, 2013. Springer Verlag.

[8] S. Gross, B. Mokbel, B. Hammer, and N. Pinkwart. How to select an example? a comparison of selection strategies in example-based learning. In S. Trausan-Matu, K. E. Boyer, and K. Crosby, M.and Panourgia, editors, *Proceedings of the 12th International Conference on Intelligent Tutoring Systems (ITS)*, number 8474 in Lecture Notes in Computer Science, pages 340–347, Berlin, Germany, 2014. Springer Verlag.

[9] S. Gross, B. Mokbel, B. Hammer, and N. Pinkwart. Learning feedback in intelligent tutoring systems. *KI - Knstliche Intelligenz*, pages 1–6, 2015.

[10] S. Gross, B. Mokbel, B. Paassen, B. Hammer, and N. Pinkwart. Example-based feedback provision using structured solution spaces. *Int. J. of Learning Technology*, 9(3):248–280, 2014.

[11] B. Hammer, D. Hofmann, F.-M. Schleif, and X. Zhu. Learning vector quantization for (dis-)similarities. *Neurocomputing*, 2013. In Press.

[12] J. Holland, A. Mitrovic, and B. Martin. J-latte: a constraint-based tutor for java. 2009.

[13] K. R. Koedinger, J. R. Anderson, W. H. Hadley, and M. A. Mark. Intelligent tutoring goes to school in the big city. *International Journal of AI in Education*, 8:30–43, 1997.

[14] A. Mitrovic, M. Mayo, P. Suraweera, and B. Martin. Constraint-based tutors: A success story. In *Proceedings of the 14th International conference on Industrial and engineering applications of artificial intelligence and expert systems*, pages 931–940, London, UK, 2001. Springer-Verlag.

[15] B. Mokbel, S. Gross, B. Paassen, N. Pinkwart, and B. Hammer. Domain-independent proximity measures in intelligent tutoring systems. In S. K. D'Mello, R. A. Calvo, and A. Olney, editors, *Proceedings of the 6th International Conference on Educational Data Mining (EDM)*, pages 334–335, Memphis, TN, 2013.

[16] B. Mokbel, B. Paassen, and B. Hammer. Adaptive distance measures for sequential data. In M. Verleysen, editor, *22th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN, Bruges, Belgium, April 23-25, 2014*, pages 265–270. i6doc.com, 2014.

[17] B. Mokbel, B. Paassen, and B. Hammer. Efficient adaptation of structure metrics in prototype-based classification. In S. Wermter, C. Weber, W. Duch, T. Honkela, P. D. Koprinkova-Hristova, S. Magg, G. Palm, and A. E. P. Villa, editors, *Artificial Neural Networks and Machine Learning - ICANN 2014 - 24th International Conference on Artificial Neural Networks, Hamburg, Germany, September 15-19, 2014*, volume 8681 of *Lecture Notes in Computer Science*, pages 571–578. Springer, 2014.

[18] B. Mokbel, B. Paassen, F.-M. Schleif, and B. Hammer. Metric learning for sequences in relational LVQ. *Neurocomputing*, (accepted/in press), 2015.

[19] T. Murray, S. Blessing, and S. Ainsworth, editors. *Authoring Tools for Advanced Technology Learning Environments*. Kluwer Academic Publishers, Dordrecht, 2003.

[20] S. Ohlsson. Constraint-based student modelling. In J. E. Greer and G. I. McCalla, editors, *Student Modelling: The Key to Individualized Knowledge-based Instruction*, pages 167–189. Springer-Verlag, Berlin, 1994.

[21] K. Vanlehn. The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education*, 16:227–265, August 2006.

[22] J. Wittwer and A. Renkl. How effective are instructional explanations in example-based learning? a meta-analytic review. *Educational Psychology Review*, 22(4):393–409, 2010.