EPFL

Thèse n° 8037

Learning Analytics for Adaptive and Self-Improving Learning Environments for Inductive Teaching

Présentée le 23 juillet 2021

Faculté informatique et communications
Laboratoire d’ergonomie éducative
Programme doctoral en informatique et communications

pour l’obtention du grade de Docteur ès Sciences

par

Louis Pierre FAUCON

École
polytechnique
fédérale
de Lausanne

2021

Acknowledgements

First, I would like to dearly thank my advisor Pierre Dillenbourg. His lab is a fantastic environment to learn and grow as a PhD student. I thank Pierre for giving me both guidance and freedom in the pursuit of my research interests. I feel lucky to have been one of his students. I also thank Florence who assisted me in all necessary administrative tasks, always with kindness.

Next, I wish to thank Tanja Käser, Michael Yudelson, Ulrich Hoppe, and Bob West for accepting to be part of my thesis jury. I thank them for the time they took from their schedules to evaluate my dissertation and participate in my thesis defense.

This thesis would not have been possible without the guidance and mentorship of many postdocs that I thank for their help and guidance. First, I thank Lukasz Kidzinski for supporting my projects when I was still learning about probabilistic models and machine learning. I thank Stian Haklev for our exceptional collaboration and his never ending enthusiasm to engineer learning technology. I particularly wish to thank Jennifer Olsen for her mentorship. She contributed significantly to my scientific knowledge and academic rigor. I also thank her for the invaluable help in the redaction of this manuscript.

I feel lucky to have been part of such a joyful lab. I thank Sina, Alexis, Thibault, Barbara, Kevin, Konrad, Hala, Aditi, Javi, Arzu, Ayberk, Kshitij, Jade, Utku, Sven, Ramtin, and many others for the good times we've had together all these years. I also want to thank all the friends that I met at EPFL. First, I wish to thank Antoine for the fascinating discussions and competitive chess games we've had during countless lunches and coffee breaks. I want to thank Konrad, Tomas, and Julian from the Effective Altruism Association and Lê, Mahdi, Mariame, Aidan, Sergei, Sébastien, Felix, and Anastasiia from the Lausanne Alignment Club. They have helped me become a better version of myself. I specially thank Lê who instilled in me curiosity and a passion for Bayesian probabilities.

Finally, I would like to thank my wife, Pin for her unconditional love and support throughout the most challenging times. I also thank my family for always being here and keeping me on the right track.

Lausanne, 5 November 2020 Louis

Abstract

The emergence of digital technology is changing education in many ways. A particularly interesting aspect of this transformation is the development of learning environments that can automatically adapt to individual students and can collect data in order to automatically improve themselves. How a learning environment should adapt to students is dependent on the pedagogical approach. In this thesis, we contribute to adaptive and self-improving learning environments that support students during inductive reasoning activities. Inductive teaching is a pedagogical approach that yields very beneficial outcomes for students, but has not received sufficient technological support. Using inductive reasoning, students infer general rules or concepts from observations. This pedagogical approach is motivated by the idea that learning outcomes will increase when students construct the knowledge by themselves.

First, we contribute to student modeling and self-improvement for learning environments. We share the results of a study that we conducted using data collected in several large classrooms. We develop a mechanism that collects data, estimates the progress of students and predicts in real-time their future progress. The goal of our approach is to aggregate the predictions to support adaptive decisions by teachers in classrooms. Moreover, we share the results of a second study with MOOC students. We use a generative model, based on a Semi-Markov Chain, to model and simulate sequences of actions taken by students on the MOOC platform. Additionally, based on observations of students from arbitrary Bayesian models, we propose and analyse a self-improving algorithm relying on Thompson Sampling optimisation that maximises students learning outcomes over time.

Second, we contribute to the analysis of inductive teaching and individual differences in inductive reasoning. We report results on an experiment with 222 students solving tasks of categorisation of images. Our study revealed that students individually differ in how they choose to classify examples based on feature differences between the examples. We define the individual differences of students as their inductive bias. This result motivates the importance of adaptivity in the selection of examples during inductive learning activities. Additionally, we analyse how much students change their inductive bias when confronted to negative feedback. We find that students are influenced by the feedback, but that this influence decays after a short period of time.

Third, we contribute multiple algorithms that constitute together the necessary components of an adaptive and self-improving learning environment for inductive teaching. Notably, we designed algorithms for a learning environment to extract representations of students' biases from data, estimate and trace individual students' biases, and to optimally select personalised examples for an inductive learning activity.

Finally, we conclude by describing the concept of probabilistic testing as a promising assessment mechanism for inductive learning environments. We provide preliminary observations and theoretical results for the use of probabilistic testing in the context of adaptive inductive teaching. In particular, by using probabilistic tests, a learning environment can estimate more quickly students' inductive biases.

This thesis includes multiple computational methods in learning analytics, student modeling, adaptive teaching, and self-improvement. We apply these methods to different aspects of learning environments for inductive teaching. When looked at closely, the process of inductive reasoning reveals an incredible epistemological depth. Undoubtedly, inductive teaching has the potential to improve educational technology and deeply benefit students.

Key words: Learning Environment, Learning Analytics, Adaptive Teaching, Self-Improvement, Bayesian Student Models, Bayesian Inference, Inductive Teaching

Résumé

L'émergence des technologies digitales change les méthodes d'éducation de nombreuses façons. Un aspect particulièrement intéressant de cette transformation est le développement d'environnements d'apprentissage qui peuvent automatiquement s'adapter à chaque étudiant et recueillir des données afin de s'auto-améliorer. La manière dont un environnement d'apprentissage doit s'adapter aux étudiants dépend de l'approche pédagogique. Dans cette thèse, nous contribuons au développement d'environnements d'apprentissage adaptatifs et auto-améliorants qui soutiennent les étudiants lors d'activités de raisonnement inductif. L'enseignement inductif est une approche pédagogique qui produit des résultats très bénéfiques pour les étudiants, mais qui n'a pas reçu à ce jour un soutien technologique suffisant. En utilisant le raisonnement inductif, les étudiants déduisent des règles ou des concepts généraux à partir d'observations. Cette approche pédagogique est motivée par l'idée que la qualité de l'apprentissage augmente lorsque les étudiants construisent eux-mêmes leurs connaissances.

Dans un premier temps, nous contribuons à la modélisation des étudiants et à l'amélioration automatique des environnements d'apprentissage. Nous partageons les résultats d'une étude que nous avons menée à partir de données recueillies pendant plusieurs cours universitaires. Nous développons un mécanisme qui permet de collecter des données, d'estimer le progrès des étudiants et de prédire en temps réel leur futur progrès. L'objectif de notre approche est d'agréger les prédictions afin de soutenir les décisions d'adaptation des enseignants dans les salles de classe. De plus, nous partageons également les résultats d'une deuxième étude avec des étudiants de MOOC. Nous utilisons un modèle génératif, basé sur une chaîne semi-markovienne, pour modéliser et simuler des séquences d'actions adoptées par les étudiants sur la plateforme de cours en ligne. En outre, sur la base d'observations d'étudiants à partir de modèles bayésiens quelconques, nous proposons et analysons un algorithme d'auto-amélioration reposant sur l'algorithme d'optimisation, Thompson Sampling, qui maximise les résultats d'apprentissage des étudiants dans le temps.

Dans un second temps, nous analysons l'enseignement inductif et les différences individuelles dans le raisonnement inductif. Nous rapportons les résultats d'une expérience impliquant 222 élèves résolvant des tâches de catégorisation d'images. Notre étude a révélé que les élèves diffèrent individuellement dans la façon dont ils choisissent de classer les exemples en fonction de leurs différences de caractéristiques. Nous définissons les différences individuelles des élèves comme le biais inductif propre aux élèves. Ce résultat souligne l'importance de l'adaptabilité dans la sélection des exemples lors des activités d'apprentissage inductif. Afin de mieux connaître ce qui influence les étudiants, nous analysons dans quelle mesure les étudiants modifient leur biais inductif lorsqu'ils sont confrontés à un retour d'information négatif. Nous constatons que les étudiants sont influencés par le feedback, mais que cette influence diminue après une courte période de temps.

Dans un troisième temps, nous proposons plusieurs algorithmes qui constituent ensemble les composantes nécessaires d'un environnement d'apprentissage adaptatif et auto-améliorant pour l'enseignement inductif. Nous avons notamment conçu des algorithmes pour un environnement d'apprentissage permettant d'extraire des représentations des biais des élèves à partir de données, d'estimer et de retracer les biais de chaque élève et de sélectionner de manière optimale des exemples personnalisés pour une activité d'apprentissage inductif.

Enfin, nous concluons en décrivant le concept de test probabiliste comme un mécanisme d'évaluation prometteur pour les environnements d'apprentissage inductif. Nous fournissons des observations préliminaires et des résultats théoriques pour l'utilisation des tests probabilistes dans le contexte de l'enseignement inductif adaptatif. En particulier, grâce à l'utilisation de tests probabilistes, un environnement d'apprentissage peut estimer plus rapidement les biais inductifs des étudiants.

Cette thèse propose de multiples méthodes algorithmiques dans des domaines tels que l'analyse de données, la modélisation de l'étudiant, l'enseignement adaptatif et l'amélioration personnelle. Nous appliquons ces méthodes à différents aspects des environnements d'apprentissage pour l'enseignement inductif. Lorsqu'on examine de près le processus de raisonnement inductif, il révèle une incroyable profondeur épistémologique. Il ne fait pour nous aucun doute que l'enseignement inductif a le potentiel d'améliorer la technologie éducative et d'être profondément bénéfique pour les étudiants.

Mots clefs : Environnement d'apprentissage, Analyse de données, Enseignement personnalisé, Auto-amélioration, Modélistation d'étudiants, Inférence Bayésienne, Enseignement inductif

Contents

1 Introduction

The rise of digital technology has transformed education in an irreversible way. Changes are continuously occurring, and there is an incredible opportunity to continue to improve education by investigating how we can leverage digital technology. A recent example of the importance of digital technology for education is the still ongoing Coronavirus disease (Covid-19) pandemic, which has forced all the universities in multiple countries around the world to transition to online teaching and, as a consequence, to rely nearly exclusively on digital technologies. Another example of this transformation is the growing number of Massive Online Open Courses (MOOCs). MOOCs have quickly become an important topic in education by attracting several millions of learners from around the world. Additionally, they have attracted the attention of the educational research community, because researchers seek to improve the learning experience of the numerous students in MOOCs and because of the large amounts of learning data that can be collected and analysed. It is fair to say that digital technology has both transformed educational practices and research.

A particular aspect of the digital transformation is the availability of computational methods that are a central theme of the thesis. The MOOC hype coincided with the emergence of Big Data and the broad use of applications of Machine Learning (ML) within educational technology. Many researchers employ novel computational methods to build models of high predictive performance to discover new insights for the learning sciences [14, 106, 221]. This thesis results from this wave of change and seizes the opportunity brought by novel computational methods.

Throughout the thesis, we will classify a learning environment as any system that participates in the interaction with students with a teaching objective. For example, a classroom with a professor is a learning environment as is a private tutor, a MOOC, an Intelligent Tutoring System (ITS), or a notebook with a pen. If digital technology is used in a classroom, then the combination of the technology and the classroom is the learning environment. The focus of this thesis is on learning environments that are at least partly digital. In this introduction we will define multiple tasks that an adaptive and self-improving learning environment must perform in order to teach well. Throughout the thesis, we aim to provide technological solutions to build digital systems that perform these tasks automatically.

A learning environment is adaptive if it interacts differently with the students based on their individual differences, such as knowledge, reasoning ability, or learning style. The goal of an adaptive learning environment is to improve teaching by providing personalisation. A challenge in this domain, which has been known for several decades, is Bloom's two sigma problem [24]. Bloom observed that students who receive individual tutoring perform on average better than 98% (two standard deviations) than students who do not. Thus, the challenge is to build automated systems that are able to teach individual students at least as well as a tutor without the high cost and small scale of human tutoring. Digital technology has brought multiple opportunities to achieve this goal in many areas of education, notably using ITSs [54, 171, 182, 243].

A second goal of the thesis is the concept of self-improving learning environment. Once a learning environment is built and used by students and teachers, there are often possibilities for improving it. A learning environment is self-improving when the mechanisms for improvement are automatic and embedded into it. Self-improvement can take many forms. In the thesis, we focus on the improvement of the adaptivity mechanism in the learning environment.

Different approaches to teaching have benefited differently from the opportunities of digital technology. For example, adaptive teaching has been implemented within ITS with a focus on mastery learning [56]. An interesting distinction can be made between deductive and inductive teaching approaches. Both with or without digital technology, the majority of teaching is done deductively. Yet, there have been examples of ITS that improved students' learning by using inductive methods [65, 219, 218]. However, these examples only aimed to adapt whether or not to use inductive activities and not to adapt the inductive activities themselves. More research in this area will benefit the study and practice of inductive teaching. As the focus of the thesis, we chose to study adaptive and self-improving mechanisms for learning environments using inductive teaching. We hope that our contributions will motivate the development and use of such technology and benefit students.

In this introductory chapter, we define the concepts of the thesis and point towards the different chapters. In Section 1.1, we describe the landscape of opportunities enabled by digital technologies. In Section 1.2, we define the key problems that adaptive learning environments have to solve. In Section 1.3, we define the notion of a self-improving learning environment. In Section 1.4, we focus on inductive teaching. Finally, in Section 1.5, we summarize the outline of the thesis.

1.1 Opportunities from Digital Technology

Countless fields of research are being revolutionised by digital technology including biology, chemistry, and theoretical physics. Research in education is not an exception to this phenomenon. This section focuses on five particular aspects of this transformation that are relevant for the different works presented in this thesis. These five aspects are the development of new user interfaces, the connectedness between learners and the learning environments, the increased scale, the possibility to analyse the learning processes through learning analytics and the possibility to automate high quality teaching through artificial intelligence.

Although the thesis focuses primarily on the aspects of learning analytics and artificial intelligence, the three other aspects are nevertheless important to take into consideration. The opportunities in all five areas that we discuss below are not independent from one another. For examples: the user interfaces are important to consider for a learning environment that seeks to connect students; the scale of a learning environment is important to collect large quantity of learning data, which enables to benefit more from artificial intelligence methods; artificial intelligence allows us to optimise user interfaces.

1.1.1 User Interfaces

With the advancements in digital technology, novel user interfaces have been developed in order to enable a wide range of new learning experiences that benefit students' learning. These interfaces include virtual reality [60], robotics [170, 105], and tangible interfaces [250]. Another type of interface is real-world computer-simulations [121, 224] that have been used in physics [202] to teach concepts such as velocity and acceleration [111] or in chemistry [118]. MOOCs are also a new type of user interface enabled by digital technology. Never before would students have imagined that the classroom teacher would have three buttons to pause, slow down, or backtrack. Finally, simple paper tests such as Multiple Choice Question can now be automatically distributed online and automatically graded with a lot less effort from teachers and teaching assistants.

This thesis does not focus on the design of improved user interfaces for education, yet it is a line of work complementary and equally important. Additionally, the author participated in the design of the educational platform FROG [92], which is used to teach live classrooms. FROG has the concept of interface at the core of its implementation. Using FROG, the teacher plans a lecture by sequencing several activities, which correspond to different interfaces for the students, such as chats, quizzes, programming interface, or simulations.

In Chapter 8 we also describe a different user interface for MCQs which changes the answering mechanism in order to collect more detailed information from students and to be able to better adapt to their knowledge states.

1.1.2 Connectedness

A second benefit of computational technologies is the increased connectedness between students to all parts of the learning environment. We define connectedness as any exchange of information. For examples, two students must be connected in some way to do a collaborative activity, or a learning environment must connect data from students to the teacher to show a dashboard describing students' performance. Notably, the work that we detail in Chapter 2 relies on being able to connect data from the whole classroom in real-time in order to analyse more precisely the data from individual students.

Digital technology led to the emergence of the field of Computer-Supported Collaborative Learning [68] thanks to the new possibility for connecting students during learning activities. Notably, one of the FROG platform's main focuses is collaboration between students [93]. This is achieved through multiple mechanisms. Firstly, automatically forming small groups of students and connecting them through a collaborative activity such as a chat, a synchronised text editor, or a brainstorming interface. Secondly, students are connected through the output of their work, which can be used in subsequent activities during the lecture (for example, through peer reviewing). Finally, when using the FROG platform, all the students are connected to the teacher through real-time dashboards that allow the learning environment to provide information to support the teacher's decisions.

With digital technology, separated parts of the learning environment are now connected in ways that were impossible before. These connections can make tasks, which required previously large efforts, instantaneous using automation. Examples of such connection between multiple parts of a learning environment can be found in the GLUE!-PS research project [184] or the orchestration workbench developed by Phiri, Meinel, and Suleman [179].

Connectedness through digital technologies also means that students do not need to be physically present to receive instruction. The field of distance learning strongly relies on digital technology to provide high quality instruction. Notably, FROG, through the use of digital technology, allows one to enhance both students' cognitive activities and social presence in online classrooms [167].

1.1.3 Scale

A third impressive transformation of learning environments is that they have allowed for education to be provided at scale. Platforms for Massive Open Online Courses (MOOCs) such as Coursera[1] or EdX[2] were motivated by the widespread access to the Internet. As of today, hundreds of universities have partnered with such platforms and have accumulated tens of millions of course registrations. One of the main motivations for developing MOOCs is the incredible number of students that can be reached. With courses counting several tens of thousands of participants, MOOCs bring the opportunity to multiply the teaching output of teachers. Beyond MOOCs, an increasing number of online learning platforms such as Khan Academy[3], Brilliant[4], or Duolingo[5] have reached millions of learners around the world.

[1]https://www.coursera.org/
[2]https://www.edx.org/
[3]https://www.khanacademy.org/
[4]https://brilliant.org/
[5]https://www.duolingo.com/

Social Networks also attract pedagogical content that reaches millions of learners every day. For example, the Youtube channel 3Blue1Brown[6], which published hundreds of videos on advanced mathematical concepts, has millions of views on each video on average.

These new possibilities led to the emergence of the research field of Learning at Scale, which specifically studies learning environments with a high ratio of learners to facilitators, such as MOOCs or ITSs [199]. The scale and reduced number of facilitators per student is made possible by the technological automation of multiple tasks necessary to the learning environments. We analyse, in Section 1.2, what some of these tasks are and contribute, in the rest of the thesis, methods to automate them.

1.1.4 Learning Analytics

An additional advantage of digital technology for education is the possibility to automatically collect and analyse data from learners. These practices are studied within the research field of Learning Analytics [14] and the closely related field of Educational Data Mining [15], which are a central theme of this thesis. Examples of data collection enabled by digital technology range from click-streams in MOOC videos [133], interactions with robots [164], eye-tracking [213], and measuring head-motion [188]. Combining these data streams led to the sub-field of multi-modal learning analytics, which aims to analyse data combined from multiple sources to better understand the learning processes [23]. As a consequence, digital learning environments can rely on data from two sources. The first is data from the usual learning activities of students as they transitioned from being done on paper to the computer. The second is data from new measurement tools. These are not required by the learning activities, but can bring an additional level of measurement, which can be beneficial to analyse and optimise the students' learning process.

Moreover, our ability to analyse the data and leverage it has been transformed. Computational methods have raised an unprecedented opportunity to improve our understanding of learning and students [106]. The prediction of students' behaviour and outcomes has become a very popular research subject. Indeed, looking into the method and results of these prediction algorithms often gives insights on the dynamics of students' learning processes [57, 247, 251]. We discuss this further in Chapter 3. Additionally, learning analytics methods can be used to improve learning environments. For example, teacher dashboards help teachers make more informed decisions in their teaching process (See Chapter 2), but also help teachers with the assessment of students [152, 197].

1.1.5 Artificial Intelligence

A last benefit from digital technology that is the most relevant to the scope of this thesis is the opportunities to improve learning environments using Artificial Intelligence (AI). We men-

[6]https://www.youtube.com/c/3blue1brown/videos

tioned previously that individual tutors can help students perform two standard deviations above the level of students who do not receive tutoring [24]. That is because the tutor very efficiently adapts to the student and is able to provide personalised teaching that maximises the student's learning gains. AI in many domains imitates human intelligence for numerous tasks. In the context of education is the research field of Artificial Intelligence in Education[7] (AIED), which includes a large corpus of research on ITSs [210].

There is no reason for an adaptive and self-improving learning environment to be limited to the capabilities of individual tutoring with a human tutor. Computational methods have shown capabilities beyond humans at games such as Chess, Go [220], or Atari [12], and experts tasks such as medical imaging [107] and mental health diagnosis [74]. This thesis aims at designing systems with the ability to adapt to students' knowledge and reasoning and teach potentially better than a personal tutor. This goal has, within the research community, not yet been fully achieved. Yet, we believe the work of this thesis is one step on this fabulous endeavour for improved educational systems.

In this introduction, we will not discuss in precise details the use of AI in education, because that is a topic that we address in the following chapters of the thesis. In particular, we design and analyse mechanisms using ML methods to provide high quality personalised teaching. Indeed, AI is a key component for adaptive and self-improving learning environments.

1.2 Adaptive Learning Environments

Adaptive learning environments, although they can be easily defined as providing personalised teaching to individual students involve a number of complex mechanisms. In this section, we decompose adaptive teaching in multiple parts: modeling students' states, providing learning activities, observing students' performance and behaviour, tracing students' state transitions and optimising the teaching strategies. Our decomposition is displayed on Figure 1.1. In this section, we explain the importance of each part and how they are connected. The decomposition helps us better understand the problem of adaptive teaching and is used as a framework to guide our contributions in the following chapters of this thesis.

The decomposition can be used to explain how a teacher adapts to the students. First, before the beginning of a lecture, the teacher will have an idea of what the students already know of the course material. Then, the teacher will provide learning activities such as listening to the lecture, working through exercises, or answering a multiple-choice test. During and after the activities the teacher will observe students behaviour and performance. For example, the level of noise in the classroom or the number of correct answers to a multiple-choice test. Based on these observations, the teacher will trace the evolution of students' knowledge and will seek to optimise the rest of the lecture accordingly.

[7]https://iaied.org/

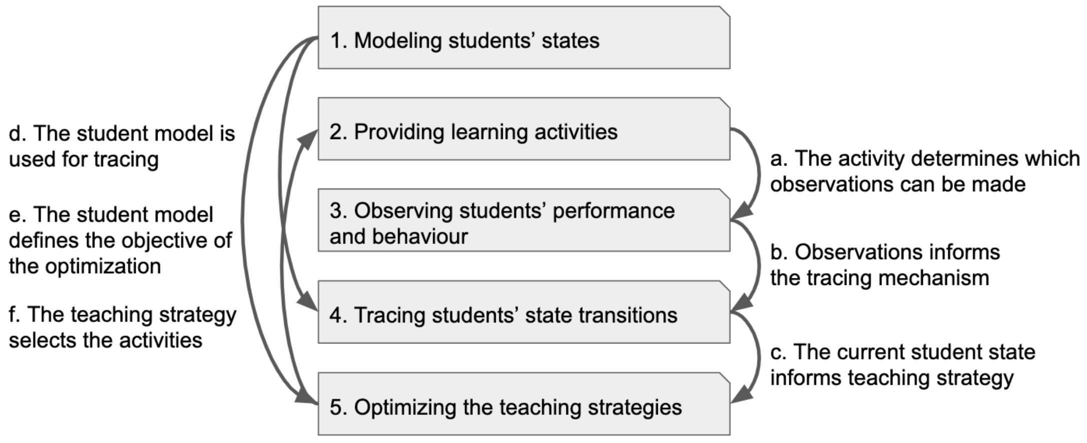

1. Modeling students' states
2. Providing learning activities
3. Observing students' performance and behaviour
4. Tracing students' state transitions
5. Optimizing the teaching strategies
a. The activity determines which observations can be made
b. Observations informs the tracing mechanism
c. The current student state informs teaching strategy
d. The student model is used for tracing
e. The student model defines the objective of the optimization
f. The teaching strategy selects the activities

Figure 1.1 – Division of the problem of adaptivity into sub-problems

1.2.1 Modeling Students' States

Self [210], in the context of ITSs, mentioned that student models are necessary for a learning environment to *care* about the students. Student models are algorithmic representations of the states of students [211]. Generally, a student state observable behaviour or performance of the student and how the student can transition to different states when interacting with a learning environment. The goal of teaching is, ultimately, to change students, and student models can be used to measure how students change. Typically a student starts interacting with the learning environment in a given initial state, and after a sufficient amount of time, the student should be in a different state. When using a student model, the learning environment will have certain expectations of what it will observe from the student given the student's current state. For example, before learning a skill, the student is expected to not be able to solve exercises, but, if learning is successful, the learning environment would expect the student to be able to solve exercises.

Student models are, in general, simplifications and not necessarily very detailed descriptions of students. A model is considered useful, not because of its complexity, but because it predicts and explains students' behaviours or performances well. In Bayesian Knowledge Tracing (BKT) [55], which has been implemented in multiple ITSs, the state of the student for a given skill is one of two possibilities: either the skill is mastered or it is not. In other models such as Item Response Theory [238], Learning Factor Analysis [38], or Performance Factor Analysis [176], the state of a student is described by a continuous value corresponding to the student's level of ability and learning for a given skill. The state of the student does not necessarily correspond to the students' knowledge.

Student models are not necessarily restricted to describe students' knowledge or ability. It is useful in multiple contexts to model other aspects of the state of the students, such as their level of effort and motivation [214], their progress through a task (see Chapter2), or their behaviour (see Chapter 3).

1.2.2 Providing Learning Activities

The learning activities are the means through which the learning environment interacts with the students. In MOOCs, the primary activities are watching videos, in-video quizzes, several kinds of assignments (multiple-choice tests, numerical values or programming exercises) and forum interactions for seeking help, providing help or discussing interesting aspects of the course. In a classroom, learning activities can be of multiple types. A few examples are lecturing, solving problems, practicing exercises, and having interactive discussions. Both instructors and researchers have worked to push the boundaries of activities that can be done in classrooms. Digital technology allows for even more possibilities as discussed in Section 1.1.1. FROG is an example of digital platform that seeks to bring richer types of activities and pedagogical scenarios into the classroom, such as arguing, brainstorming or experiential activities [92]. The learning activities that a learning environment is able to use are, of course, a very strong factor in the quality of the teaching it provides.

1.2.3 Observing Students' Performance and Behaviour

Observing students is a necessary condition to being able to personalise education because personalisation requires knowing the individual characteristics of a student. The learning environment could observe only the performance of students (for example, correct or incorrect answers to questions), or it could observe other aspects of the student behaviour as they are relevant to adapt the instruction [6]. We mentioned in Section 1.1.4 the multiple sources of data that can be collected from students that are enabled by digital technology. It is important to note that collecting data from students in itself does not imply optimal teaching. In the context of an adaptive learning environment, observations are used to enable adaptivity. In particular, the design of learning activities must not exclude data collection mechanisms (Figure 1.1a). It is difficult to imagine a tutoring lesson where the tutor does not ask questions (data collection mechanism) to assess the student's knowledge and understanding.

Throughout the thesis, as we design algorithms to model student behaviour and optimise teaching strategies, we pay a particular interest in the data that the learning environment observes from students. Additionally, in Chapter 8, we will focus on a particular data collection mechanism tailored for inductive reasoning (see Section 1.4).

1.2.4 Tracing Students' State Transitions

Another task that adaptive learning environments must solve is the tracing of students' state transitions. As we previously mentioned, student learning is characterised by a change of state. Tracing state transitions implies that the system must evaluate how learning activities change students' states and then infer the new state of the students by potentially using additional observations. This is a central aspect of an adaptive learning environment. Tracing happens in accordance to two mechanisms: the description of the student model (Figure 1.1d) and the available observations (Figure 1.1b).

One of the most used models for tracing students' knowledge is BKT [55]. At every step the student has a chance to transition from non-mastery to mastery and the model estimates the probability that the student has changed state based on the observation of correct or incorrect answers. Several models proposed in this thesis approach knowledge tracing in a similar manner. In particular, in Chapter 3, we provide a model that estimates the individual behaviour of students in a MOOC, and in Chapter 6, we provide a Bayesian Model for tracing students' inductive reasoning approaches.

1.2.5 Optimising the Teaching Strategy

The optimisation of a teaching strategy can be considered to rely on two mechanisms. The first one is the tracing. To be able to make optimal choices for an individual student the learning environment will greatly benefit from high quality estimates of the state of the student (Figure 1.1c). The second aspect of the optimisation is more challenging. The learning environment must choose learning activities that will balance between two different goals: collecting information about the student's state and influencing the transition of the student towards more desired states (Figure 1.1f). Some activities only collect information about the student (for example, filling a form at the beginning of an online course) and other learning activities only influence the state of the student (for example, reading a book in the library). Interestingly, some activities can help both goals simultaneously. This has been the case for example with in-classroom quizzes for test-enhanced learning [198].

This type of optimisation is well explained with the concept of Partially Observable Markov Decision Processes (POMDP) that have been shown to compute efficient teaching strategies in an educational context [190]. Finally, optimisation requires an objective. We will thus assume that the learning environment has preferences about the possible states of the students (Figure 1.1e).

1.3 Self-Improving Learning Environment

Improving learning environments over time can be a great gain for education in general. Although we measure improvement of a learning environment in terms of learning outcomes of students (including the time it takes to achieve these learning outcomes), improvement can occur in many ways. Several possibilities could be to improve the pedagogical content, the learning activities, or the parameters of an adaptive teaching strategy.

Outside of digital technology, one can think of multiple examples of pedagogical systems that improve themselves. For example, a professor accumulating higher quality learning material

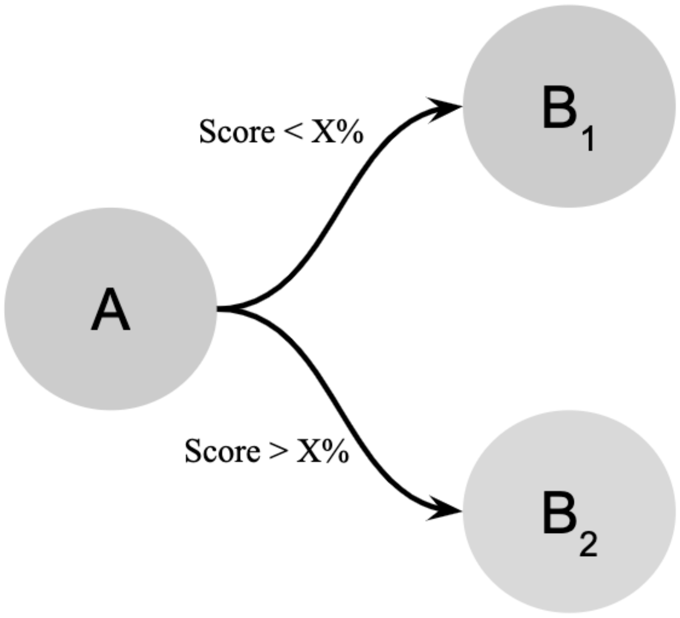


Figure 1.2 – Example of self-improving problem

every year, or a university that makes use of student feedback to enhance the curricula. For a digital learning environment to be self-improving, it needs to contain mechanisms that evaluate different teaching strategies and over time improve the quality of the teaching strategies that it uses.

Self-improvement differs from adaptivity. While interacting with a student, Alice, an adaptive learning environment learns about Alice and becomes better at teaching Alice. On the other hand, while interacting with Alice, a self-improving learning environment slightly improves its teaching strategy (this is done by analysing the impact of its own teaching on Alice) and becomes better at teaching the next students Bob and Charlie.

Figure 1.2 provides an illustrative example. An adaptive learning environment might decide that after activity A, students should do either activity B1 or B2, depending on their performance on activity A. For example, using the comparison of the score with a threshold X, a previous test with 24 students gave some evidence that $X = 33\%$ was a better choice than $X = 66\%$. At that point an adaptive learning environment will only use the threshold of $X = 33\%$, while a self-improving learning environment will continue to test different values of the parameter X and aim to converge to an optimal value after sufficient exploration.

The Exploration-Exploitation Trade-off

Self-improving systems often rely on the ML concept of the exploration-exploitation trade-off [32]. Exploration consists of the system making decisions that allows it to acquire more useful information, and exploitation consists of it using the collected information to maximise the expected rewards. In the context of education, the goal of a self-improving learning environment is to select good pedagogical choices while measuring the efficiency of these choices and exploring enough different possibilities to not miss on the opportunity to discover better teaching strategies. Always selecting the choice that seems optimal according to the data collected so far removes the opportunity to discover better choices. Self-improvement and how it applies in an educational context is discussed in more details in Chapter 4 and Chapter 7 of this thesis.

1.4 Inductive Teaching

Adaptive teaching is already a thoroughly researched topic. However, countless teaching practices have been developed and some have benefited less than others from the opportunities of personalised instruction. In this short section and in several chapters of the thesis, we define inductive teaching and discuss several interesting aspects of it. As we found that this practice did not receive much attention in the research on adaptive teaching while it indisputably deserves it, we chose inductive teaching as the main focus of this thesis.

Induction is the practice of inferring a general rule based on the observation of examples. Induction is often opposed to deduction. Using deduction, one starts from general rules and draws conclusions about the examples. These different types of reasoning are both important and lead to different teaching approaches. When teaching deductively the teacher will first teach the general rules and theories, then will guide students to apply them on specific examples. With the inductive teaching approach, students are first presented with examples and the conclusions and must guess what general rule would explain the given conclusions.

An example of induction consists of showing the 16 shapes on Figure 1.3 and telling a student that the five highlighted examples are all the squares. The students are then given the task of finding explanations for this categorisation. Here the explanation "Squares have 4 right angles" is insufficient because some of the shapes fit that rule but are not highlighted. Hopefully, students would be able to guess the correct rule: "Squares have 4 right angles and 4 sides of equal length". The deductive approach to this exercise would be to first tell students that "Squares have 4 right angles and 4 sides of equal length" and then to highlight all the squares among the 16 shapes.

Inductive teaching is motivated by the idea that students will more deeply understand and remember the knowledge they construct by themselves [95]. Additionally, with inductive reasoning, students not only learn the topic of interest but also practice discovering new information by themselves [102]. These reasons have made induction an interesting topic of study for the learning sciences and motivated the focus it is given in this thesis.

Studying how humans reason inductively has been a challenge for ML. The ability of children to learn new concepts from a very small number of examples is still unchallenged by today's AI algorithms [61]. For example, a 12-year-old child who sees two horses for the first time and is

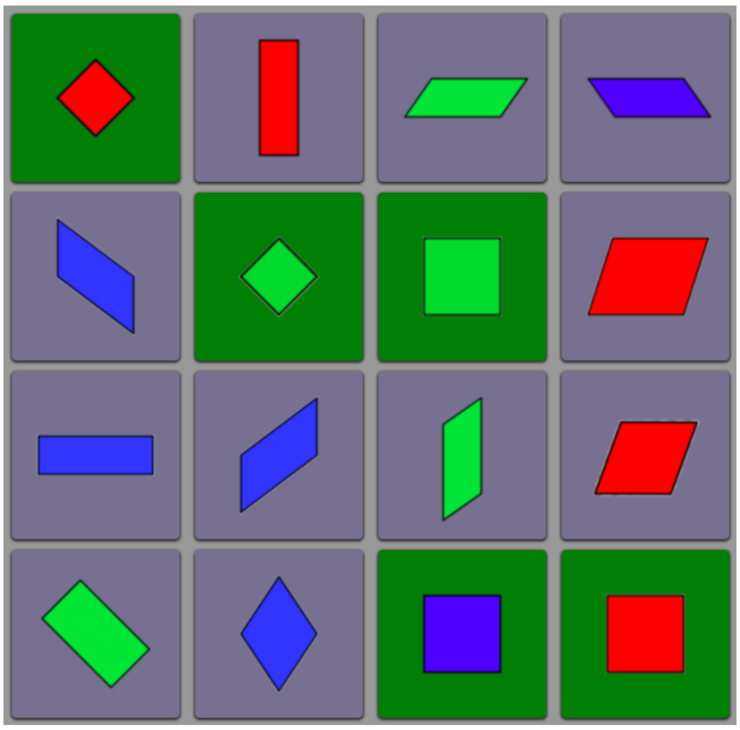


Figure 1.3 – Example of inductive exercise. Out of 16 geometrical shapes 5 squares have been highlighted in green.

told the name of these animals would be able immediately to easily recognise other horses. In ML, the concept of learning from a small number of examples is called Few-Shots or One-Shot learning and has been a compelling field of research [203]. As most ML algorithms fail to learn with so few examples, researchers are studying how humans perform such inferences.

Bayesian Inference is one of the most promising approaches for explaining human inductive capabilities [232]. To justify this modeling step, it is interesting to note that most induction is probabilistic. Indeed, in general, multiple rules will be satisfactory explanations for the examples observed. The multiplicity of explanations of observed examples makes inductive reasoning uncertain. Although there is uncertainty, the confidence in the induced conclusions can very well be very high. To illustrate this point, we can observe that the explanation to the categorisation on Figure 1.3 could have been "squares have 4 right angles and 4 sides of equal length and must be green, red, or blue". A student inferring this rule will later be wrong when encountering a yellow square, yet nothing in the examples allows one to conclude with certainty that this second proposition is incorrect. The examples provided might be sufficient to be quite confident that the color does not matter in the definition of a square, yet, they are insufficient to be absolutely certain of that conclusion.

We thoroughly define and analyse the mechanisms of inductive teaching and inductive reasoning in Chapters 5, 6, and 7 of this thesis. Although several ITS for inductive teaching have been developed [65, 218, 217], they only personalise the decision of using or not an inductive approach, not the inductive learning activity itself. In this thesis, our goal is to design algorithms that would allow one to build learning environments that would adapt the inductive activities to individual students. Additionally, we seek to make these learning environments self-improving as they will improve the adaptive mechanism after multiple interactions with students. Such adaptivity for inductive teaching was missing before our contribution.

1.5 Thesis Outline

The thesis is structured as follows. In **Chapter 2** we describe a learning analytics method for predicting student progress in real-time during activities in classrooms. In **Chapter 3** we analyse ML models for predicting and simulating students' behaviour. We apply such models in the case of simulating MOOC students' behaviours and compare our simulations with a dataset of 500,000 students. In **Chapter 4**, we further define the concept of self-improvement for a learning environment and contribute algorithmic foundations. In **Chapter 5**, we focus on inductive teaching, specifically generalisation from examples. We report multiple results concerning how students generalise depending on different features of examples and how their reasoning is influenced by feedback. In **Chapter 6**, we contribute a Bayesian Model for tracing students' individual differences and flexibility during inductive reasoning tasks. In **Chapter 7**, we further define the model of inductive reasoning and contribute algorithms for optimal teaching, adaptivity and self-improvement. Finally, in **Chapter 8** we analyse an assessment mechanism specifically tailored to our model of inductive reasoning.

The thesis contains work previously published by the author. Chapter 2 of the thesis contains work published under [79]. Chapter 3 contains work published under [77]. Chapter 4 contains work published under [76]. Chapter 5 and Chapter 6 contain work published under [78]. We duly note that the author of the thesis is the main author in all the work directly reused in this thesis.

2 Learning Analytics

One central aspect of this thesis is the concept of learning analytics, which consists of collecting and analysing data from students. This practice can lead to the development of data-driven educational technology or improve our understanding of the learning process [15, 14]. In this chapter, we report on a study that we carried out and led to the development of a learning analytics tool for lectures in classrooms, which include learning activities on digital devices. Specifically, our tool estimates and predicts in real-time the progress and completion rates of students during learning activities in classrooms [79].

Let us consider a simple sequence of two learning activities. First, learners individually solve a problem. Second, the teacher forms pairs of learners who produced different solutions in the first activity and asks them to solve a new problem. This scenario is a typical collaborative script [67]. Now, what happens if the teacher gives the students 10 minutes to complete the individual activity, but after 10 minutes, only 90% of the students have produced a solution? This is a frequent problem. The teacher is then confronted with a difficult decision: extend the time allocated for the first activity or force the transition to the next activity with some students being paired randomly. This decision constitutes a dilemma: have 90% of the learners wait (knowing that idle learners will often engage in distracting off-task activities) or continue without the 10% of learners who do not have a solution. The teacher could make a more informed decision if a system could provide information such as “the completion rate of the first activity will be around 96% in two more minutes and 100% in nine more minutes." This example illustrates the goal of this chapter: *predicting completion time to optimise the timing of transitions between activities.* Two key concepts of our contribution are: **classroom orchestration**, the management and adaptation of learning activities in real time to account for events as they occur in the classroom [66, 127, 183], and **orchestration dependency** between two activities, A and B, if A sets up logistical conditions for B (e.g., A produces data used by B, or if A is an individual activity and B involves the whole classroom simultaneously).

We aim to contribute to classroom orchestration by investigating how we can predict student progress in real time during an activity to support teachers’ decision-making in their classroom.

Specifically, we are interested in predicting the progress that students will make over a period of time and the time of completion of the activity based solely on the actions that have been taken in the activity, without any previous information. In this way, our model does not have to be calibrated on a new activity or group of students. Our models were tested on the data gathered across 12 activities run in four large lectures. By using 12 datasets, we can test the algorithms across a range of groups and activities to make stronger claims on the generalisability of our results. Moreover, because we are using data collected during class activities as opposed to in the lab, our data will follow more authentic completion patterns. The range and authenticity of the data allow us to investigate the impact that different classroom features have on the progress and completion predictions.

To resonate with the framework that we presented in the introduction (Section 1.2), we model the state of students as their progress and completion rate during a learning activity. The digital platform provides learning activities and also automatically collects progress data from students, and we predict the evolution of students' progress over time (the main focus of this chapter). These predictions are used by the teacher to better optimise the use of time during the lecture.

The sections of the chapter are organised as follows. In Section 2.1, we discuss related work supporting and modeling classroom timing. In Section 2.2, we present the formulation of the problem with the datasets and context presented in Section 2.3. In Sections 2.3.3 and 2.4, we present our proposed estimators and results, respectively.

2.1 Related Work

2.1.1 Timing Instruction

As our simple example in the introduction illustrated, within formal education settings, the amount of time for instruction on a certain topic is limited [66, 115], so teachers must continuously consider how to balance the time spent on any activity so it is most productive for the class as a whole. However, in the classroom, it can be difficult for teachers to monitor and track the state of all students. It is important to balance giving students enough time to work on an activity so that it is useful for their learning with other activities that need to be accomplished. Normal classroom cues, such as the noise level, may not be appropriate indicators of classroom completion because some students may be discussing the task while others are engaging in off-task behaviour. These factors make it difficult for teachers to estimate the appropriate time to switch to the next activity. Learning analytics could be used to help teachers make these timing decisions.

Within classrooms, there are different classifications of academic learning time at which different interventions can have different effects [87, 115]. At the highest level is the *allocated* (planned for) *time* that the teacher has provided for the activity. In the classroom, external constraints can lead to not all of the allocated time being used for instruction [66]. The actual time used for an activity in the classroom is the *instructional time*. Different individual factors of the students, such as their motivation [148] or prior knowledge, can change how students interact with the activity, as well as the use of instructional time, which is measured through the *engaged time*. Finally, although a student may still be engaged with a task, given the nature of the task, the student may no longer be productive, leading to a reduction in the *successful and productive learning time*. When considering activity transitions in the classroom, the teacher is working to maximise the engaged and productive learning times of the students.

Across these different levels of classroom timing, researchers have found that an increase in instructional time has a positive effect on learning [84]. When the instructional time is limited, students have a reduced ability to develop knowledge integration of complex topics [46]. Because teachers do not necessarily have control over the amount of time they have with students during the day, it is important to optimise the instructional time that they do have. This can be done by decreasing non-instructional time, such as transitions between classes or activities, which, through support, has been shown to increase time on task [36, 101, 256]. Learning in the classroom can also be supported by influencing the engaged time and successful and productive learning time. For example, when the time limits for an activity are explicitly announced, students increase the number of tasks completed without reducing their accuracy [194, 238, 239]. However, students still do not all work at the same pace and may become disengaged when they complete a task.

Within the literature, the primary way that student pacing is addressed is through individual personalised learning [229, 255]. In personalised learning, students work at their own pace and on topics that benefit their learning. Although time is still limited in the classroom, each student can potentially use this time optimally. However, within personalised learning, students often work individually, but once dependencies are introduced with other students in the class, timing matters again. For example, when students are working collaboratively or as a whole class, they depend on group members or classmates to be ready for the activity. Even on an individual level, such as with peer grading, there may be dependencies where students rely on other students to be ready before they can continue to the next activity. These dependencies can lead to off-task time, where some students finish early and have to wait for other students to be ready. Even giving students a productive task to work on during this time may still have a negative impact [69]. To reduce this wait time, teachers need to be able to make informed decisions about when to move to the next task that are optimised for the entire class.

2.1.2 Progress Awareness Tools

To support teachers in optimally deciding when to move to the next task, there needs to be support for classroom orchestration. The definition of classroom orchestration includes the management and adaptation of learning activities in real time to account for events as they occur in the classroom [127, 183]. Although learning activities are often planned in advance, variability occurs in the classroom that must be accounted for in real time. Two examples of this variability are groups needing to be changed because a student is absent and students taking more or less time on an activity than planned. In these cases, the teacher plays a central role in assessing the current state of the classroom and adapting the learning activity to fit the current needs of the students, given the external constraints of the learning environment (e.g., limited time, physical classroom environment) [66, 70].

When helping teachers make decisions about when to have an activity transition, current orchestration systems and teacher dashboards account for activity progress in one of two ways: tracking student progress and tracking the passage of time. One metric that teachers can use to understand when students should switch to the next activity is students' progress in the current activity. Only when a student has completed the current task can they move on to the next. The progress that students are making is not always visible to teachers. Visualisations can be used to increase teachers' awareness of student progress [222, 248]. One method of visualising progress is to use mastery grids or heat maps [141]. Mastery grids and heat maps display a grid of students and different activities. Each square represents the progress that a specific student has made on a specific activity. This visualisation allows the teacher to quickly assess how far students or a class have progressed through a set of activities and where they may be struggling. These visualisations are often used after the fact for reflection [41, 244] but have also been used in real time for teachers to track student progress to better understand which students may need more teacher support [162].

Although both student progress and the passage of time provide different types of information to the teacher, they have not often been combined into one visualisation. In some tools, the teacher can track student progress and time through separate visualisations [151]. However, in these systems, the teacher is still responsible for interpreting the visualisations separately and then combining the information. Additionally, these visualisations do not provide a prediction, meaning that the teacher can only make decisions based on the current state of students, which is often too late. In this chapter, we aim to develop models that combine the aspects of time, progress, and prediction. In other words, through the information provided by the model, the teacher could determine the expected progress that the class will have made in three minutes (or any given amount of time) or how long until the class has reached 95% progress (or any given progress), which is not possible with current methods.

2.1.3 Modeling Student Behaviours

In terms of analysing time, the focus has primarily been on finding patterns in student habits or in predicting completion. When students are working in a self-paced environment, such as when learning in a MOOC, they can determine their own schedules and when to work. What time during the day, during the week, and across the whole course students choose to work has been found to be related to learning outcomes and performance [27, 82]. However, these models primarily attempt to find patterns in how students use their time rather than predicting timing. Other research has focused on the overall goal of whether students will complete a course [57, 62]. Using features of student activity while learning, such as clickstream data or prior test scores, we can predict student completion of a class or program. Although this research relates to completion, it relates more to the concept of *whether* a student will reach completion rather than *when* they will reach completion.

Real-Time Predictions

Prediction algorithms are one of the most popular methods in learning analytics and educational data mining. Prediction methods allow us to infer a predicted variable based on other variables in the data. They have been used to predict future events, such as student completion/dropout [57, 62] or knowledge tracing and formative assessment [81, 160, 176], or to predict variables that may not be feasible to collect in an educational setting, such as engagement [18] or affective states [73].

Using prediction methods in real time, interventions can be put in place before it is too late to have an impact on the student's learning. A well-known application of predictions in real time is Bayesian Knowledge Tracing [55] to track student knowledge while using an intelligent tutoring system. Based on this knowledge tracing, new problems for the student can be chosen to match their skill level [242]. Making predictions in real time has unique challenges in that the data used for the prediction has to be easily accessible. In many systems, log data is used even when predicting more complex phenomena. For example, response times can be used to predict engagement [18], student actions can be used to detect student misuse of software [13], and affective state can be inferred from conversational cues [73]. Using log data allows models to be developed in a non-intrusive way while still being able to be used in real time.

Real-time predictions require appropriate data processing and restrictions to support the algorithms and students' models. For example, predicting next week's dropout in MOOCs can only be done with the data collected for the current week and the previous weeks, but never with the data from future weeks [230]. A solution to this problem has been to use transfer learning [29, 251]. Transfer learning allows us to train models using data from the final weeks of previously finished MOOCs and use this acquired knowledge to correctly predict students' behaviour for the final weeks of the currently running MOOC. Another proposed solution uses proxy labels in real time from already collected data to train algorithms to predict future data [251]. Our approach in this chapter faces the same difficulty in its real-time predictions. The estimators we propose cannot compute optimal parameters due to unknown future data. We apply methods similar to proxy labelling by training our algorithms solely on data collected during the current activity in the classroom (see Section 2.2.3 for a more detailed description of our method of handling the real-time aspect of our data collection and prediction mechanism).

Aggregating Classroom Information

Before a visualisation can be developed to be displayed to a teacher, we first need to be able to accurately model classroom behaviours. In the areas of research on educational data mining and learning analytics, models of students have a predominant place. Many of the methods focus on finding different patterns in the data that can be used to provide teachers with more information and to inform their decisions in their classroom [14]. When used to make visualisations, the outputs can be interpreted by humans to make decisions. The majority of models that are developed target modeling individuals, which can be limited for making classroom-level decisions. When information is aggregated, it may be to show the collective student knowledge to get a more complete picture of what the class knows as a whole [222]. Additionally, aggregated data can be used to see overall class patterns, such as average activity and actions when video watching [247] or the behaviours that students engage in across a course [112]. Although these methods can provide useful information about student behaviour and actions over time, they cannot make predictions based on students' previous and current actions.

2.2 Formulation of the Problem

The problem addressed in this chapter consists of designing estimators to predict the future progress of students during activities in the classroom. Specifically, we focus on tasks that can be divided into discrete steps, such as quizzes, as opposed to open-ended tasks, such as essays or code writing. The estimators described in Section 2.3.3 below can be used in any learning environment that automatically extracts estimates of students' progress in real time. In technology-enhanced classrooms, this extraction can easily be integrated into most learning software.

Before describing the estimators of students' progress in Section 2.3, we first discuss the key aspects of the problem. In Section 2.2.1, we define the concepts of progress, progress rate, and completion and explain the context in which our estimators will be implemented and used. In Section 2.2.2, we point to the main factors and challenges influencing the design, in Section 2.2.3 we discuss the specificity of real-time predictions, and in Section 2.2.3 we explain how data from the whole class can improve estimations of individual students' progress.

2.2.1 Definition of Progress and Completion

We define the notion of *progress* using a value between zero and one that estimates how far through the task students have advanced. A value of zero corresponds to the student not having started the activity, while a value of one corresponds to the student having completed the activity. Reaching *completion* in an activity means that the student has completed all required subtasks and, thus, will wait until the teacher transitions to the next activity. In this sense, we can think of the progress as the percentage of the activity that has been completed if completion is binary, being either done or not. Progress as a percentage can be interpreted in two different ways. It can be evaluated according to the percentage of the subtasks completed or the percentage of the time already passed out of the total time required. Counting the number of subtasks completed is easily implemented within a learning environment. On the other hand, we must predict the amount of time students will need so we can measure their progress in terms of time. Progress in terms of time needed can be a more interesting measure for the teacher but unfortunately cannot be directly known. For this reason, we prefer a definition in terms of percentage of subtasks completed. In line with this definition of progress, we define *progress rate* as the speed at which a student's progress is increasing over time. We assume that a student cannot make backward progress; that is, once a student has completed a subtask, it remains completed.

We also assume that student progress cannot be actively and precisely measured at all times. Instead, progress is reported based on milestones reached. When students are working on an activity, progress may be made both inside and outside the orchestration system. For example, on a multiple-choice quiz for mathematics, students may use paper to work out the problem before selecting the solution in the system. As they work on the problem on paper, they are making progress on the problem that cannot be precisely measured by the system. With our estimators we restrict ourselves to working with discrete measurements of student progress that the system is able to provide. As a consequence, we focus on learning activities that can easily be divided into subtasks. Whenever a student completes a subtask, the system collects their student ID, time, and progress. For every student with ID s, we then have a set of tuples $(t_{s,0}, p_{s,0}), (t_{s,1}, p_{s,1}), \ldots, (t_{s,n}, p_{s,n})$. We define the progress at any time t as the linear interpolation from equation (2.1), where $t_{s,0}$ is the time when the first log is received, which corresponds in our case to a progress $p_{s,0}$ of 0; t is any time during the activity; and $t_{s,k}$ and $t_{s,k+1}$ are two consecutive times at which logs are received with the respective progress of $p_{s,k}$ and $p_{s,k+1}$:

$$p_s(t) = \begin{cases} 0 & \text{if} \quad t_0 \geq t, \\ p_{s,k} + (p_{s,k+1} - p_{s,k}) * (t - t_{s,k}) / (t_{s,k+1} - t_{s,k}) & \text{if} \quad t_{k+1} \geq t > t_k, \\ p_n & \text{if} \quad t > t_n. \end{cases} \tag{2.1}$$

2.2.2 Primary Challenges

In an ideal world, all students would complete the activity at the same time and work at a steady pace. In classrooms, this does not happen because both endogenous and exogenous pitfalls influence when individual students complete an activity. Challenges in predicting student progress arise due to these individual differences between students. In this section, we present some of the primary challenges to predicting progress, including different start times and varied progress rates between subtasks.

Variability in Students' Start Times

A first challenge for predicting student progress is that students do not all start the activity at the same time for a number of reasons, such as being distracted or taking longer to start their computer or sign in. For example, Figure 2.1 shows two distributions of the delays of students for starting an activity. The right distribution shows a majority of students starting the activity within 20 seconds of the teacher making the activity available. On the other hand, the left distribution shows a significant proportion of the students starting the activity one or even two minutes after the activity is made available. These examples demonstrate the range of starting situations that can occur depending on different parameters in the classroom. Variability in start times is more of a challenge for teachers, leading to more variability in completion times and a larger variance in student progress.

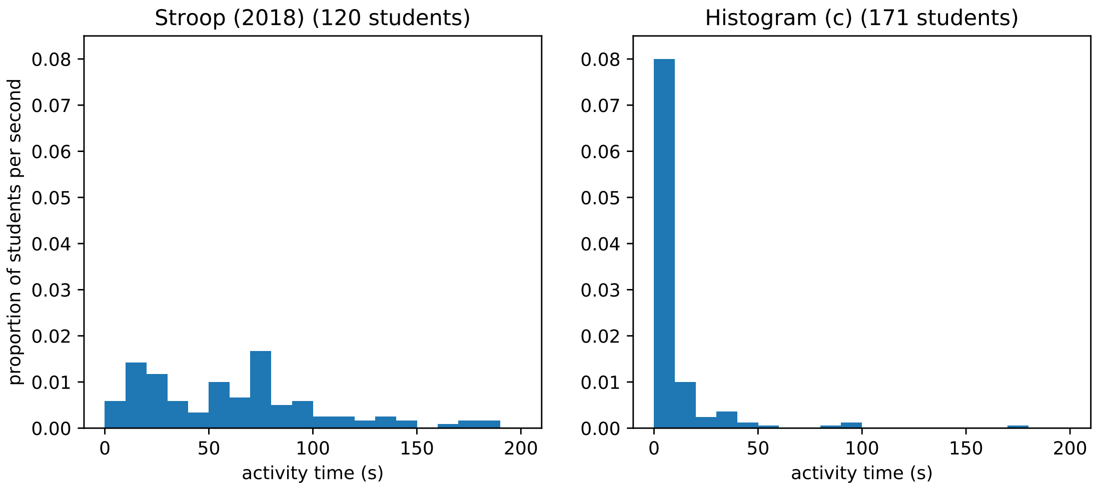


Figure 2.1 – Examples of distribution of students' starting delays for two activities in different classrooms.

Variability in Student Progress Rates

A second challenge is accounting for differences in student progress rates, which are often a consequence of individual differences. For a wide range of possible reasons, different students will require different amounts of time to complete a given activity. Knowing this time distribution of the classroom is a challenge for teachers. A good way to quickly evaluate the spread of the distribution of individual progress rates is to measure how much more time the slower students need as a percentage of how much time the faster students use. For this we can choose, for example, to use the 25% and 75% quartiles of the time distribution to correspond respectively to our estimate of the time needed by faster and slower students. Using this measurement, our data from twelve courses (described in more detail in Section 2.3.2) has three courses where the slower students needed more than 50% more time than the faster students, four courses where the slower students needed between 30% and 50% more time than the faster students, and five courses where the slower students needed between 20% and 30% more time than the faster students. The two histograms in Figure 2.2 show two examples of activities that have different spreads. The distribution on the left corresponds to one of the courses with a spread of more than 50% according to our proposed measure, and the activity on the right has a spread of less than 25%. These individual differences in progress rate again pose a challenge for the teacher to decide how much time to allocate to an activity when students do not all need the same amount of time.

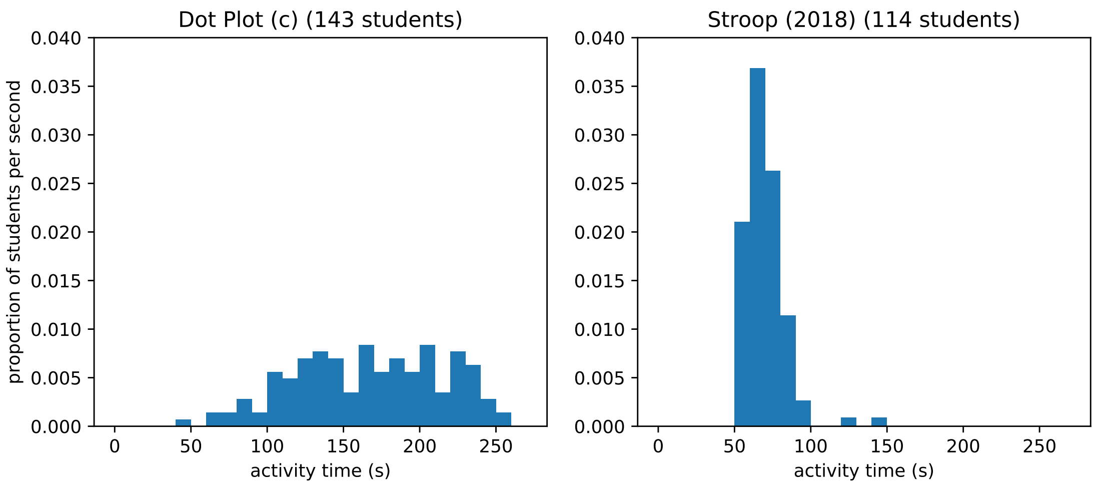


Figure 2.2 – Examples of distribution of the time students take to complete two activities in different classrooms.

Imprecision in Expectations of Time Distribution Between Subtasks

A third challenge comes from the lack of knowledge about the proportion of time that each subtask will consume out of the total time. In the simplest situation, each subtask has an equal proportion of the progress if there is no prior knowledge about the tasks, but, in most cases, each subtask does not take the same amount of time to complete. For example, Figure 2.3 shows the distribution of the proportion of time each student used for each question of a six-question quiz. We can see on this graph that the average time spent on each question varies significantly between the students and between the questions. This quiz is an example of a default progress assignment, with each question receiving an equal amount of progress, that fails to correctly capture the time students really spend on each subtask. In this example, the naive assignment would assume approximately 17% of progress for each question, while we can see from the figure that question 1 had an average time proportion above 25% and the sixth question of less than 10%.

From our experience in classrooms, we identified several factors that can make students' progress rate vary during the activity. One factor could be the skills and preferences of students, which will modify their progress rates accordingly. Additionally, it is not uncommon to design a quiz where questions become harder over time and, thus, require more time to be solved.

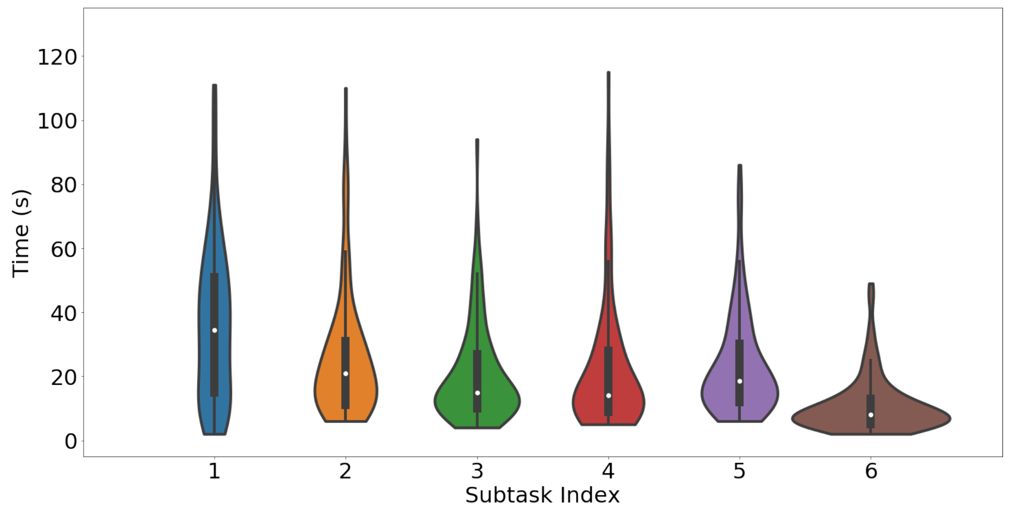


Figure 2.3 – Distribution of proportion of the time used for a six-question quiz for an activity that lasted approximately 260 seconds.

Another factor is that students are getting used to the task they are doing, which will increase their progress rate (in Figure 2.3, we see a general downward trend in the proportion of time required by students for each subtask). Finally, in class the teacher will also interact with students either to explain a difficulty or to make them aware of time passing. These interventions are not often recorded but can affect students' progress rates. In general, it is not surprising if not all the subtasks of an activity require the same amount of time to complete. This poses a challenge for estimating and predicting student progress because it makes it difficult to extrapolate the time required by future subtasks from the time used on the previous subtasks.

2.2.3 Real-Time Predictions of Future Progress

To predict students' future progress, estimators make predictions in real time, implying that at any given time, the estimators must be able to make predictions based solely on the data collected until this particular time. Thus, the estimators have to work with a varying amount of data throughout an activity. At an early stage of an activity, students will have produced only a limited number of logs, but at a later stage, a number of students will have already finished and others will have produced a larger number of log messages. The problem of the prediction for an individual student is illustrated in Figure 2.4. The figure shows five black dots representing five pairs of time and progress $(t_{s,i}, p_{s,i})$, as defined previously. This example shows an activity with 10 subtasks, so the expected log messages received from the students will be $0, 0.1, 0.2, \dots, 0.9,$ and 1. In this example, so far, the student has only completed 40% of the activity so has generated logs only up to $p_{s,n} = 0.4$. Another input used by the predictions is the current time, represented on the figure as a vertical black line. We call this the time of prediction and use the notation t_{pred} in the rest of the chapter. The estimators must predict students' progress from the latest received log message to the current time t_{pred} and from the current time to completion, which may be several minutes. Figure 2.4 (left) shows as dashed lines possible predictions using interpolations of the student's progress curve.

The predictions of current and future progress cannot be based solely on the student's logs as input. Figure 2.4 (right) illustrates this point. The blue dashed line shows a situation in which interpolation from only the sequence of logs completely fails to realistically predict the progress made between the time of the last received log, $t_{s,n}$, and the current time of prediction, t_{pred}, thus also rendering the prediction of future progress incorrect. Indeed, both the current time of prediction t_{pred} and the progress steps expected from the student matter to the prediction of future progress. The green dashed line shows a prediction that also uses the current time and the expected progress step (0.1 for the example). The information from the five log messages collected is not enough to correctly estimate the student's progress. An equally important piece of information is that no log message has been received between the time of the latest log, $t_{s,n}$, and the time of prediction, t_{pred}.

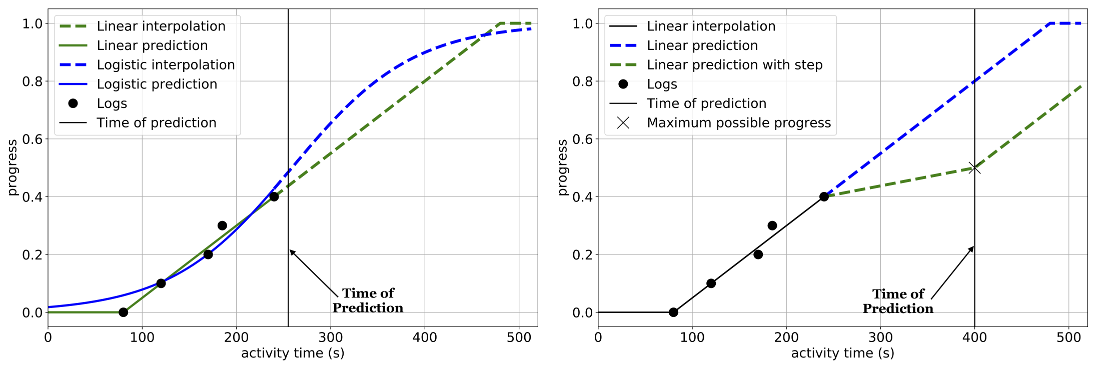


Figure 2.4 – Examples of prediction of a student's future progress using linear interpolation (green, left) and logistic curve fitting (blue, left) and prediction of a student's future progress after a delay without observing any progress from the student (right).

As part of the real-time aspect of the problem, we also introduce the notion of range of prediction, which is the amount of time in the future for which the estimators compute predictions. In practice, the range of prediction can be arbitrarily long, but predictions should be expected to be less accurate further in the future. In Section 2.4, we use fixed prediction ranges because they are necessary for evaluating our proposed estimators. Furthermore, the limited prediction range is also forced for the evaluation of the estimators because no data was collected beyond the moment when the teacher decided in class to interrupt the activities.

Interpolation Method

The small number of messages received for each individual student can be interpolated using several models. Figure 2.4 (left) shows, for example, a possible interpolation using a linear (constant) progress rate (green) or using a logistic curve fitted to the student's progress logs (blue). From the data collected on 12 courses described in Section 2.3.2, we could observe that the curves of individual students' progress are well approximated by linear curves. As we have seen from the example in Figure 2.3, the subtasks do not usually take exactly the same amount of time. However, no pattern, such as the increasing speed followed by decreasing speed of a logistic curve, could be found. Thus, it was natural to make no prior assumptions on students' progress rates and distribution of time among subtasks. This is why we decided to use linear models. Furthermore, we also investigate in this work other estimators that will learn the evolution of students' progress rate and the distribution of time among subtasks from the classroom data collected in real time without prior assumptions.

Using Whole-Class Data

Given the variance in students' start times and progress rates, it is likely that they will progress through a task in a very asynchronous manner. However, information from students who are further ahead in the activity can be used for students that have not yet reached the same point. In addition to a student's past performance being used to predict their progress, the progress from the rest of the class can be used to improve the prediction in some circumstances. For example, for a student who just started, no informed prediction can be made because we do not have any information on their progress rate. However, using the progress that other students have made through the activity, we can make an informed prediction for this student if we assume some sort of similarity with earlier students.

2.3 Methods

2.3.1 Educational Context

The data we used in our analyses was from active-learning activities conducted using the FROG platform [92]. FROG is a web-based platform for designing and executing orchestration graphs, which range from simple sequences of activities to rich integrative collaboration scripts in classrooms. With FROG, students can engage in learning activities across whole-class, collaborative, and individual activities with the ability to share data between activities and use the data to adapt learning activities and make collaborative groupings. The teacher can orchestrate the learning activities in real time using dashboards and orchestration controls to advance students through the lesson. On the student side, each planned activity is displayed on students' personal devices only when that activity is active. All interactions that students or teachers have within FROG are logged and can be used for the real-time dashboards. We focused on activities conducted using the FROG platform because it collected data at the interaction level with recorded timestamps and progress through an activity (discussed in more detail below), giving us the information needed to track progress.

Because FROG supports a wide range of learning activities, not all activities produce the same types of data, and the information that is logged for a student depends on the activity in which they are engaged. To log progress, we restricted the activities to types that had natural divisions into subtasks, as mentioned above. For example, a quiz can be divided into multiple questions, or a problem set can be divided into the individual problems. Each time a student completed a subtask within the activity, we logged this progress in addition to when the student started the activity (i.e., progress value of zero). The prediction methods presented in this chapter are not restricted to the FROG platform but do require data of this nature to be collected, where the progress of each student is logged with a timestamp throughout the learning activity. Using this individual data for each student, we can then aggregate individual progress and completion rates (i.e., progress value of one) to provide an overall class progress and completion rate along with the predicted future trajectory of both measures (see Figure 2.5 for an example of how this could be displayed).

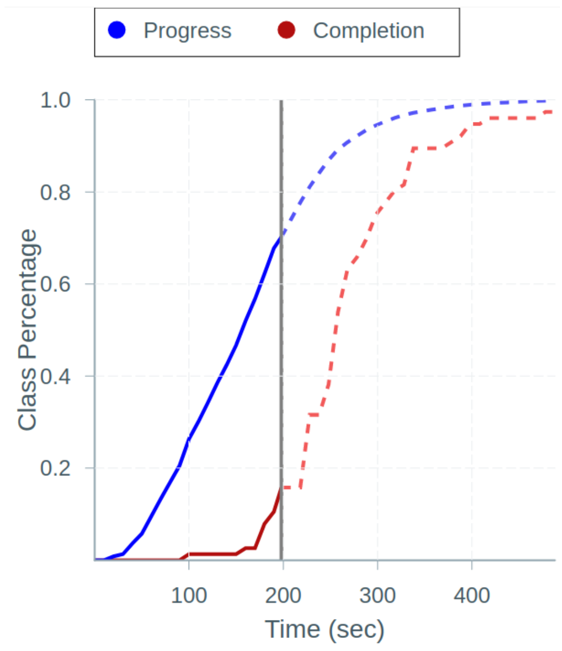


Figure 2.5 – Real-time dashboard of aggregated student progress and completion rates on the FROG platform. The vertical line is the current time; the blue line indicates the average progress of the students and the red line the proportion of the class that has completed the activity; the dashed lines are predictions of future progress and completion rates.

2.3.2 Datasets

All of our datasets come from two different courses in 2018 and 2019 in which teachers used FROG to support learning activities. The first course was a statistics course for bachelor's students. The course had about 600 students divided into three sections. We used progress logs for two activities, each given once to each section. The activities consisted of two quizzes concerning different data representations (dot plot, boxplot, and histogram). While answering the quiz, the students were also interacting with a data visualisation tool that aimed to help them explore and understand the different representations (see Figure 2.6). The two quizzes were taken by each section during their two-hour lecture. These activities are referred to as A1 to A6 in Table 2.1, and the three groups are referred to as (a), (b), and (c).

The second course was given in both 2018 and 2019 for bachelor's students in computer science. The classes had about 120 students, but attendance was often low. Both years, during three different lectures, the students did three experiential activities that aimed to teach concepts of cognition and user interfaces. The three activities were Genealogy Puzzle, Stroop Effect, and Train Ticket Simulator. Genealogy Puzzle is an activity for experiencing the effects of cognitive load. Students had to solve logic problems based on family trees of increasing difficulty, necessitating more effort and working memory. In the Stroop Effect activity, students had to answer a series of very quick questions identifying a word (of a colour) as the text changed colour. Finally, in the activity Train Ticket Simulator, students used four different user interfaces to order three train tickets. These six activities are referred to as A7 to A12 in Table 2.1.

The datasets from these classes were selected because they provide a range of activities and include students from different institutions and disciplines. For each activity, we had a dataset from more than one class, allowing us to consider variability between students. Moreover, for each class, we had data from more than one activity so we could account for the variability between activities. Because the datasets came from a class activity and not a designed experiment, no personal information was gathered from the students. All of the students were informed before participating that their data may be used for research purposes and were asked for consent. If consent was not provided, the data was deleted after that activity for that participant.

Table 2.1 shows the number of students who participated, the number of progress log messages received, the number of subtasks for each learning activity, and the length of each activity. Note that three questions were removed from the activity Dot Plot following the lecture given to the first group. This does not affect our analysis because the 12 activities are considered separately and no analysis is made between the repeated activities. Furthermore, for some activities, the number of log messages exceeds the product of the number of students and the number of subtasks. This is because extra log messages are triggered when the activity is activated by the teacher and changing an answer to a previously completed subtask triggers a repeated log message. This does not interfere with the mechanics of the predictions that we describe in Section 2.3.3.

2.3.3 Estimators

In this section, we describe several different estimators we have implemented to predict student progress in real time. The different estimators address the specific challenges of the problem that we presented above. The first models, constant progress rate (CPR) and average progress rate (APR), address the challenge of different start times and progress rates of students, while the models local progress rate (LPR) and weighted progress rate (WPR) aim to

Table 2.1 – Datasets of Progress Logs Collected. For twelve activities, numbers of students, log messages, subtasks and total time.

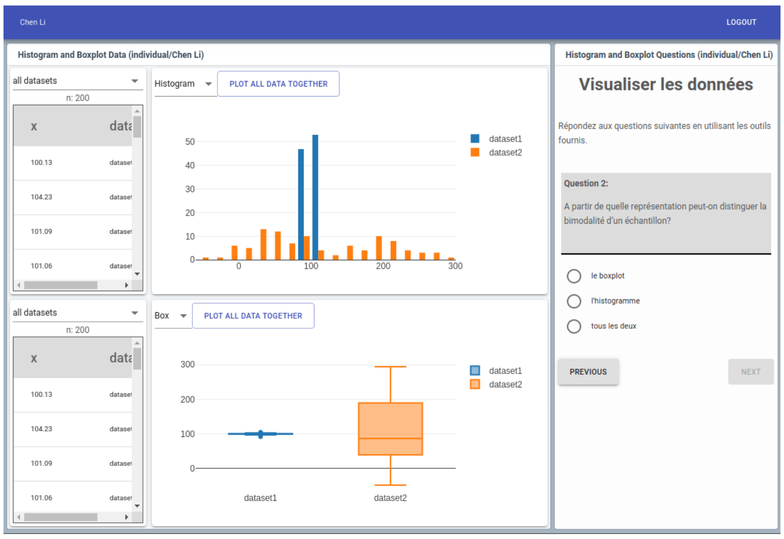
	Activity	Students	Logs	Subtasks	Total Time (s)
A1	Dot Plot (a)	241	2891	8	664
A2	Histogram (a)	227	2538	6	355
A3	Dot Plot (b)	137	1181	5	255
A4	Histogram (b)	149	1485	6	227
A5	Dot Plot (c)	163	1490	5	266
A6	Histogram (c)	171	1756	6	264
A7	Genealogy (2018)	78	704	8	386
A8	Stroop (2018)	120	2459	20	217
A9	Train (2018)	64	782	12	562
A10	Genealogy (2019)	70	667	8	425
A11	Stroop (2019)	85	1853	20	141
A12	Train (2019)	84	1060	12	636

solve the weight challenge. Finally, the models APR and WPR also use the whole class's data to inform predictions made for individuals.

CPR Model

Our first model assumes that students are making progress at a constant rate through the subtasks of a particular activity. The rate of progress can, however, be different for each student. This assumption leads us to use a linear interpolation to predict the progress of the students after estimating their progress rate. At the time t_{pred} when we are making a prediction, we estimate both the current progress and the APR for each student to compute the linear interpolation. From a student's logs, $(t_{s,0}, p_{s,0}), \ldots, (t_{s,k}, p_{s,k})$, with $t_{s,k} < t_{pred}$, we compute the progress rate to be the amount of progress made by the student divided by the time the student has been working. This computation is shown in equation (2.2), where $\hat{r}_s(t_{pred})$ is the estimated progress rate of student s, $(t_{s,0}, p_{s,0})$ is the first log entry received, and $(t_{s,k}, p_{s,k})$ is the latest log entry received before the current time at which we are making the prediction. Second, we compute the current estimate of the student's progress at the current time t_{pred} following the formula in equation (2.3). It is important to note that the true value of the student's progress at time t_{pred} is not known because the system only measures the discretised progress from the latest log of the student, $(t_{s,k}, p_{s,k})$. From this, equation (2.3) estimates the current progress, $\hat{p}_s(t_{pred})$, by adding to the latest progress received, $p_{s,k}$, the minimum of the full quantity of the progress assigned to the next subtask and the quantity of progress we expect the student to have made assuming the estimated progress rate given by $\hat{r}_s$:

$$\hat{r}_s(t_{pred}) = (p_{s,k} - p_{s,0})/(t_{s,k} - t_{s,0}), \tag{2.2}$$



Visualiser les données

Répondez aux questions suivantes en utilisant les outils fournis.

Question 2:

A partir de quelle représentation peut-on distinguer la bimodalité d'un échantillon?

- le boxplot
- l'histogramme
- tous les deux

Figure 2.6 – Screenshot of the statistics quiz activities (on the right) with the data visualisation tools (on the left).

$$\hat{p}_s(t_{pred}) = min(p_{s,k} + (t_{pred} - t_{s,k}) * \hat{r}_s(t_{pred}), p_{s,k} + step). \tag{2.3}$$

Using both the prediction of the progress at time t_{pred} and the estimated progress rate, we predict the progress for all time $t > t_{pred}$ with the linear function described in equation (2.4). This linear function has a slope of $\hat{r}_s$ and a value of $\hat{p}_s(t_{pred})$ at $t = t_{pred}$:

$$\hat{p}_s(t) = \hat{p}_s(t_{pred}) + \hat{r}_s * (t - t_{pred}). \tag{2.4}$$

The above formulas require collection of at least two logs to be able to make predictions for a student. In the particular case of students who have not yet completed a subtask (only one data point has been collected), the estimated default progress rate is always zero, which leads to a prediction of $\hat{p}_s(t) = 0$ for all $t \geq t_{pred}$.

LPR Model

The assumption that students maintain a constant progress rate is certainly too optimistic, which motivates us to consider this model of non-constant progress rates. The LPR model makes predictions based only on a fixed number κ of the latest progress logs for each student. This method supports a change of progress rate over time, where the most recent progress logs are better indicators of future progress than older progress logs. In the model, κ is a parameter for which we can choose any value $2 \leq \kappa \leq m$, with m being the number of subtasks. Large values correspond to a model very similar to the CPR model, while smaller values only account for the student's current progress rate but are also more subject to noise in the estimation because the model uses fewer data points. We compare values of κ from $\kappa = 2$ to $\kappa = 5$.

APR Model

The APR model is a variation of the CPR model. This model uses the average progress rate measured over the data collected from the whole class (see equation (2.5)) to improve estimates for students' progress rates, especially for students who have only completed a small number of subtasks. Indeed, the CPR model has several limitations, such as assigning a progress rate of zero to students who have not yet completed a subtask or being sensitive to noise in the estimate of the progress rate for students with a small number of subtasks completed:

$$\bar{r} = \frac{1}{n} * \sum_{s}(\hat{r}_s). \tag{2.5}$$

For this purpose, the APR model computes a new estimate of the progress rate for each student. It does so using the formula described in equation (2.6), where $\hat{r}'_s$ is the new progress rate of the student. In this formula, the parameter λ is the number of log messages already received from the student. The progress rate of a student with only one log message, for whom we cannot estimate the progress rate, is estimated as $\hat{r}'_s = \bar{r}$. The weight of the APR then decreases for students with more log messages:

$$\hat{r}'_s = (\bar{r} + (\lambda - 1) * \hat{r}_s)/\lambda. \tag{2.6}$$

WPR Model

As mentioned in previous sections, one challenge for the predictions is that the estimated progress steps for each subtask of an activity do not correctly estimate the proportion of time students need. Such discrepancies could lead to imprecision in the progress prediction. The WPR model proposes to solve this challenge by using data collected over time from the whole class to measure the proportion of time that students allocated on average to each subtask. This data allows us to transform the progress curves of students to account for the uneven time needed by each subtask. An advantage that the WPR model has over the LPR model is that it can account for more uneven progress changes instead of assuming that a student's progress rate will remain similar to that in their previous subtasks, as is assumed in the LPR model. The WPR model also assumes that the progress rate changes of the students are correlated, which is likely to be the case.

The first step for the WPR predictions is to compute a weight profile that represents the weight of each subtask of an activity. Given the constraints of the problem, the weight profile has to be dynamically estimated from incomplete data during the learning activity in the classroom. In other words, until a student finishes all subtasks, the complete weight profile cannot be known. However, the dynamic estimation cannot be accurate with too little data and, in particular, benefits the most from collecting log data for students who have fully completed the activity. For a student s who has generated the sequence of log messages $(t_{s,0}, p_{s,0}), (t_{s,1}, p_{s,1}), \dots, (t_{s,n}, p_{s,n})$, the proportion of time spent on each subtask is computed using equation (2.7). From the equation, we see that the proportion of time used cannot be computed for subtasks that the student has not yet completed ($k > n$), so these subtasks are assumed to require a proportion $1/m$ of the student's time, where m is the number of subtasks. For the subtasks that have already been completed, the proportion of time needed for each subtask can be directly computed from the student's progress logs:

$$\hat{w}_{s,k} = \begin{cases} \frac{t_{s,k} - t_{s,k-1}}{t_{s,n} - t_{s,0}} \times \frac{n}{m} & \text{if} \quad 1 \le k \le n, \\ \frac{1}{m} & \text{if} \quad n < k \le m. \end{cases} \tag{2.7}$$

To compute the weight profile using data from students who have not yet completed the activity, we assume that the proportion of time spent on the subtasks that have not been completed yet will be as assigned by the original weighting (this corresponds to a value of $1/m$ as in equation (2.7)). Furthermore, because students who have fully completed the activity give us more information about the true weight profile, we give more importance in the computation to the students who have completed more subtasks. This is controlled by a parameter p, which balances the amount of importance that is given. Equation (2.8) shows how the proportion of time used by each student is aggregated to estimate the true weight profile. In the equation, $\hat{w}_k$ is the aggregated weight given to subtask number k. α_s is the importance in the aggregation given to student s. α_s is computed as an increasing function of the student's progress and controlled by a parameter p. In Section 2.4, we analyse the quality of the weight profile estimation over time based on the parameter p used to average the data from students:

$$\hat{w}_k = \sum_{s \in S} \alpha_s \hat{w}_{s,k} = \sum_{s \in S} p_{s,n}^{p} \hat{w}_{s,k}. \tag{2.8}$$

Figure 2.7 is an example of an evolution of the estimated weight profile at different times during an activity (we chose to display every interval of 30 seconds). We see in the figure that the approximated weight profile estimates (blue) change over time to more precisely align with the final whole-class weight profile (orange). The final weight profile is the weight profile computed from the whole class after the end of the activity. How quickly the weight profile is

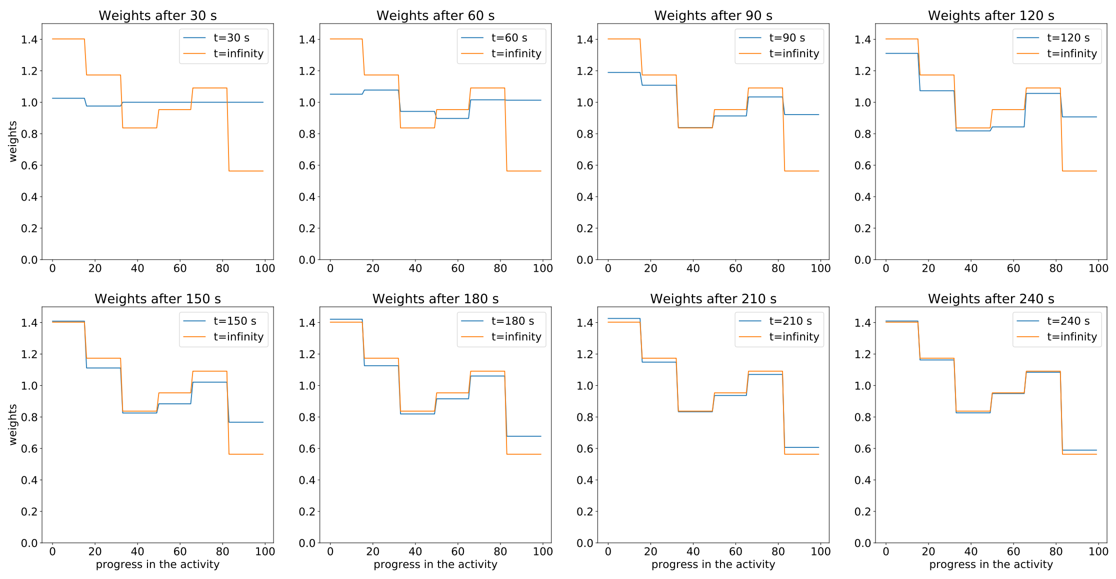


Figure 2.7 – Computation of the weight profile over time. The blue line shows the estimated weight profiles at different times by steps of 30 seconds. The orange line is the actual target.

correctly approximated is crucial for the WPR estimator because progress predictions will be inaccurate if the estimated weight profile is not similar to actual student behaviour.

The WPR model itself consists of several steps. First, based on the collected data at a given time, an approximation of the weight model is computed as outlined above. This model is used to adjust the proportion of progress assigned to each subtask of the activity from the initial teacher estimation during planning. Then the CPR or APR model is applied to the transformed data. Finally, the progress predictions are turned back to the original scale to be compared to the actual logs collected and to be shown to teachers in the scale that they choose.

2.3.4 Analysis

To analyse the different estimators, we specify a time of prediction (t_{pred}) and a range of prediction (Δ_t) for each of the 12 courses. The value of t_{pred} corresponds to a time in the class when the estimators have to make a prediction. The value of Δ_t corresponds to how far in the future the estimators are predicting students' progress. In this work, we evaluate and compare our estimators at different times t_{pred} and ranges of time Δ_t. For each prediction point, the estimators receive only the data collected before t_{pred} and must predict student progress at every future time between t_{pred} and $t_{pred} + \Delta_t$.

For a fixed time t_{pred} and range of prediction Δ_t, we compute predictions with each of the above-mentioned estimators of the interpolated progress curves for every student based only on measurements received before the time t_{pred}. To evaluate the quality of the predictions,

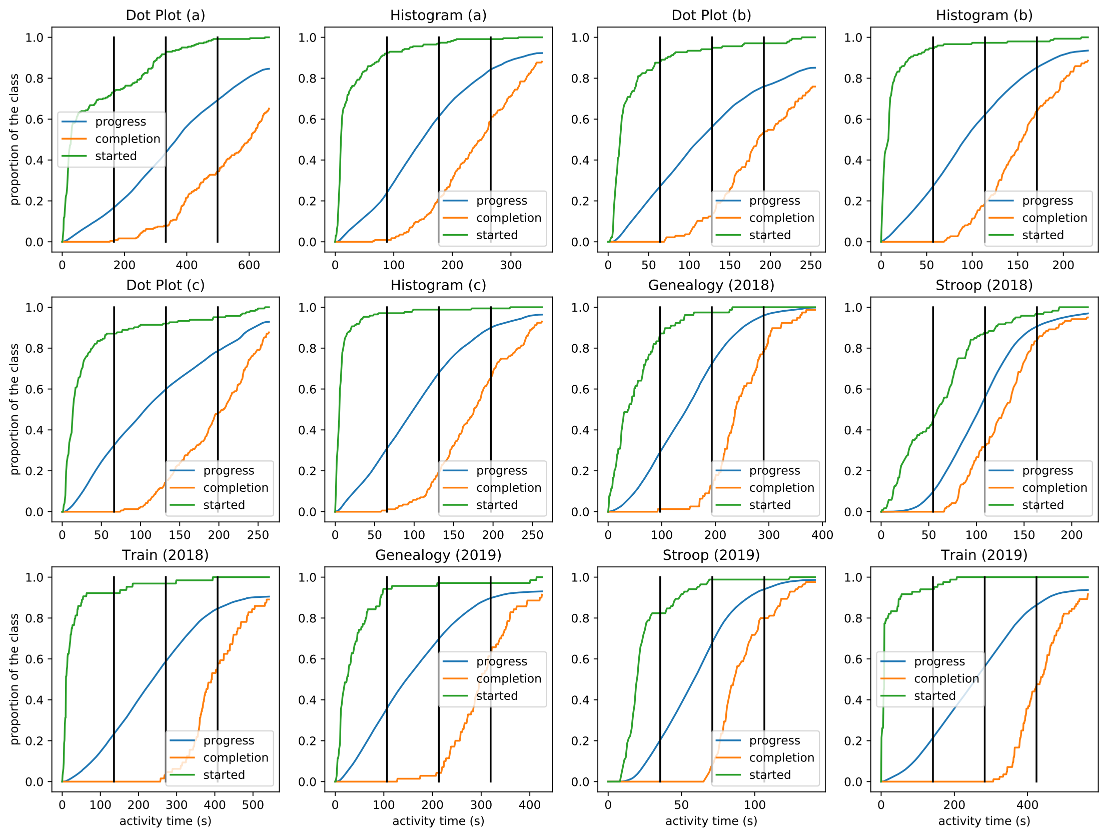


Figure 2.8 – State of each of the 12 classrooms in terms of the delay of students joining the activity (green), the average progress (blue), and the proportion of completion (orange) over time. Each graph has three vertical lines representing the times at 25%, 50%, and 75% of the full time of the activity.

we measure the root mean square error (RMSE) for each student by averaging the squared difference between the interpolated progress of the student and the predicted progress at every second. RMSE was chosen because it is a widely adopted error measure for algorithms such as linear regressions [177], which are quite similar to the estimators we are presenting in this chapter. Equation (2.9) shows the formula we used to compute the prediction error for each individual student. The sum includes the squared difference between the predicted progress and the real progress estimated for every second of time between $t = t_{pred}$ and $t = t_{pred} + \Delta_t$:

$$RMSE_{t_{pred},\Delta_t}(\hat{p}_s, p_s) = \sqrt{\frac{1}{\Delta_t}\sum_{t=t_{pred}}^{t_{pred}+\Delta_t}(\hat{p}_s(t) - p_s(t))^2}. \tag{2.9}$$

Because the accuracy of the prediction is different based on the amount of data collected, we evaluate and report on the predictions of the different estimators after 25%, 50%, and 75% of the time of the activity has passed. At 25%, a majority of students have not yet completed the activity and only a limited amount of data has been collected. On the other hand, at 75%, a majority of students have already completed the activity and others have made significant progress. Figure 2.8 shows for each of the 12 activities the aggregated progress and proportion of completion of the class over time, but also the proportion of students who have started the activity. These are important to consider when analysing the prediction results of our different estimators. For example, at the 25% prediction time for all 12 of the activities, only a very small number of students have completed the activity, so we should not expect the weight profiles estimation to be sufficiently accurate.

For the individual prediction errors measured and reported in Section 2.4, students who had already completed the activity were not included because they had already reached completion, so prediction of their future progress is trivial. Furthermore, we also did not expect the estimators to make predictions for students who join the activity after the specific time of the prediction. These two factors explain why the number of students varies depending on the time in the activity (25%, 50%, or 75%), as can be seen in Tables 2.2 and 2.3 in Section 2.4.

We compare the four previously described estimators: CPR, APR, LPR, and WPR. For LPR we used several values of the parameter κ. Again, the number κ corresponds to the number of most recent log messages that are considered in evaluating the progress rate. We tested all values of κ from two to four, which was decided based on our average number of subtasks in an activity because values larger than four would become equivalent to the CPR model. The LPR models with the different parameters are referred to as LPR2, LPR3, and LPR4 in Table 2.2 in Section 2.4.

For each activity and each time (25%, 50%, and 75%), we performed a repeated measure ANOVA to compare the prediction error of our estimators. When the repeated measure ANOVA was significant, we additionally performed a post hoc analysis using paired t-tests comparing the CPR model to each of the other models in terms of the RMSE error for each student. At each time of prediction we also performed the same statistical tests on the data from the 12 activities concatenated to give an idea of the overall performance of the estimators. However, more variance in the error measure is due to the differences between the activities rather than the differences between the predictions, as can be seen from Table 2.2 in Section 2.4. Finally, we also compare the estimates of the weight profile for different values of the estimator parameter.

So far, we have only explained how several estimators can be implemented to predict the progress of individual students. Some of these estimators use data from the whole class. Another interesting and possibly useful task is to predict the average progress of the class. This prediction allows us to build dashboards displaying aggregated information about the classroom, such as the one in Figure 2.5. This prediction could be made directly, but in our situation, because we developed several estimators to predict individual students' progress, another approach is available. In the following section, we evaluate how averaging individual predictions can be a good estimate of the future average progress of the classroom. These estimates are particularly useful for communicating information to teachers in large classrooms.

2.4 Results

The results we describe in this section are threefold. First, we analyse the optimal parameter for the weight estimation algorithm to balance the estimate between fast and slow students. Second, we compare the estimators on individual predictions. Finally, we compare the estimators for prediction of the average progress of the whole class. Section 2.4 contains our results on the dynamic estimation of the weight profiles. Section 2.4 compares the four estimators we described in Section 2.3. We report results of statistical tests for predictions made at different times during the activity, and we compare the different estimators and analyse the prediction errors.

The results of all of the comparisons are displayed in Table 2.2. For the aggregated data, we report on the fit of the data when considering aggregated progress and completion rate over the whole class in contrast to individual students. The average error rate of the estimators is shown in Table 2.3. Even though our work has been focused on individual students, the error of the aggregated predictions is a good indicator of the usefulness of our estimators in the classroom.

Evaluation of Dynamic Estimations of Weight Profiles

As mentioned in Section 2.3.3, the estimate of the weight profile used by the WPR model can be controlled by a parameter p that balances how much of the data of the first students who finish the activity influences the estimate of the subtask weights. We compared the quality of the estimate of the weight profile over time during an activity for different values of p. The main difficulty in estimating the weight profile comes from differences between the first students completing the activity and the rest of the students because the first estimates of the weight profile will be based on the data of the first students who complete the activity. Figure 2.9 shows the quality of the estimates over time for all 12 activities, depending on the choice of p. We can see that using too high a value for p leads to overfitting to the behaviour of the fastest students because data is collected about later steps of the activity only for the fast students at first, which is not completely similar to the rest of the class. This is shown in particular for the activities Histogram (a, b, and c) and Genealogy (2018). Otherwise, we see that higher values of the parameter usually lead to a faster estimation of the weight profile. For example, this can be seen for the activities Dot Plot (a and b) and Stroop (2018). Based on the overall performance for the different parameters, we decided to use $p = 2$ in our predictions.

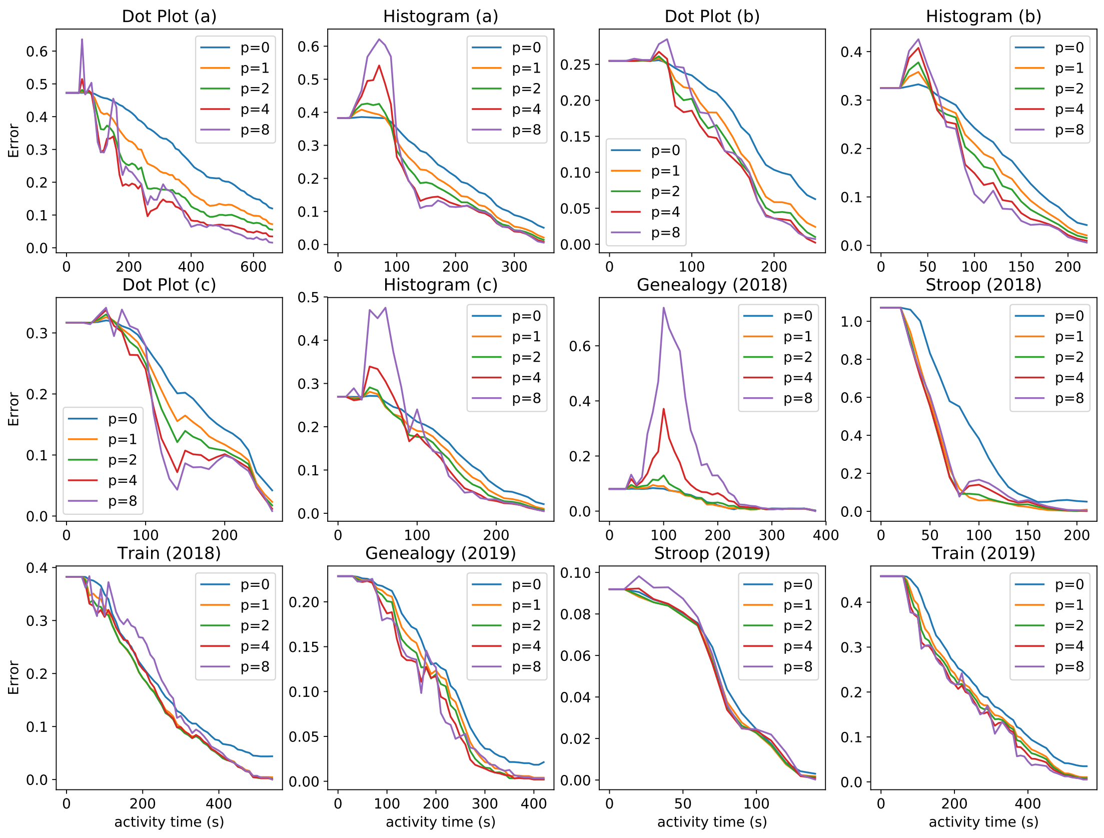


Figure 2.9 – Evolution of the error rate for estimating the weight profile over time for a range of parameters for all 12 activities. The parameter p controls the importance given to faster students in the estimates of the weight profile.

Evaluation of Individual Predictions

The results displayed in Table 2.2 show that for the predictions at 25% of the time, the repeated measure ANOVA shows a significant difference (p-value < 0.05) for 8 of the 12 activities (A1, A2, A4, A6, A7, A8, A11, A12). Additionally, the repeated measure ANOVA shows a significant difference between the estimators when aggregating the error measure from all 12 activities. The post hoc analysis using a paired t-test comparison between each estimator and CPR shows that APR, LPR3, LPR4, and WPR are all significantly different from CPR overall (p-value < 0.01), with lower RMSEs. We note that LPR3, LPR4, and WPR only improved the error rate by a few percent, while APR reduced the error by nearly 30%. Post hoc analysis on each of the eight activities shows that APR was significantly different from CPR, with APR performing better. Also, all of the models were significantly different from CPR for A8, again with lower RMSEs than CPR. Additionally, we also observe that LPR2 was significantly different for four activities from CPR, with a better performance for A8 and A12 and a worse performance for A1 and A6. Finally, the results show that WPR was significantly different from CPR for six activities, four better and two worse.

At the 50% time point, the repeated measure ANOVA is significant overall and for six of the individual activities (A3, A6, A7, A8, A9, and A11, p-value < 0.05). For the overall measure, the estimators LPR3, LPR4, and WPR differ significantly from the CPR estimator (using paired t-tests, p-value < 0.01), with the CPR estimator having a lower performance. WPR has the lowest average error of all six estimators at this time for the overall measure. As with the 25% time, we again observe that the estimators APR, LPR2, LPR3, LPR4, and WPR perform significantly differently than CPR for predicting the progress on A8 (p-value < 0.01), again with CPR performing worse. LPR2 also has a significantly different error rate on activities A7, A3, and A6, with a lower error rate for A7 and a higher rate for A3 and A6 than for CPR. WPR has a significantly different error rate than CPR on A11, with WPR having a lower error rate, but APR has a significantly different error rate than CPR on A3, with the error rate being higher for APR.

At the 75% time point, the repeated measure ANOVA is significant on the set of all the activities and also for activities A1, A2, A3, A9, and A10. The paired t-test over all the activities shows that APR performed significantly differently than CPR on average, with APR performing worse, and APR is the estimator with the highest average error rate. LPR2 and LPR4 also performed significantly differently than CPR, with LPR2 performing worse, while LPR4 performed better. The post hoc analysis shows that the APR estimator has higher prediction errors than the CPR estimator, with this difference being significant (p-value < 0.05) for activities A1, A2, and A3. LPR2 and LPR3 also have significantly different prediction errors than CPR on activity A1, with the CPR errors being lower. The t-tests comparing CPR to the other models for activities A9 and A10 were not significant.

For our analysis, we compared the baseline CPR model to our APR model, which uses the average progress of the whole class to estimate the progress rate for students with few data points. We found that 25% of the way through the activity, the APR model outperforms the CPR model on most of the activities, but the opposite is true after 75% of the time has passed, in which case the APR model performs worse on three of the activities. This result indicates that at the beginning of the activity, the APR is a good predictor for the progress rate of other students, while it is not a good predictor for students who start later in the activity. Students who start the activity after 50% or 75% of the time has already passed are likely to be very different from students who began the activity right away. Thus, the APR for other students of the class is not a good predictor for their progress rate.

Second, we compared models that addressed the variability in subtask time, the LPR and WPR models, to the baseline CPR model. The LPR model is based on the assumption that the rate at which students make progress on an activity will evolve over time. From this assumption, the model we proposed computes progress rate based only on recent logs. For this model, we compared three different variations that used a progressively larger number of previous log messages for each student. One activity that performed consistently better for the LPR models was Stroop (2018), which performed better for all three types of LPR models for the 25% and 50% time points. The Stroop (2018) activity was uniquely different from the other activities, where the main point of the activity was to complete each task as quickly as

Table 2.2 – Mean Prediction Error for Predictions at 25%, 50%, and 75% of the Time for Each Activity and Averaged over All the Activities

		25% time						
		CPR	APR	LPR2	LPR3	LPR4	WPR	*n*
All*		0.273	**0.197****	0.271	**0.270****	**0.271****	**0.267****	1395
A1*	Dot Plot (a)	0.322	**0.272***	0.331**	0.323	0.322	0.331**	190
A2*	Histogram (a)	0.355	**0.210****	0.357	0.356	0.355	**0.352****	214
A3	Dot Plot (b)	0.281	0.277	0.285	0.281	0.281	0.282	129
A4*	Histogram (b)	0.313	**0.195****	0.317	0.314	0.313	0.310	145
A5	Dot Plot (c)	0.232	0.239	0.233	0.232	0.232	0.231	150
A6*	Histogram (c)	0.270	**0.146****	0.285**	0.273**	0.270	0.268	165
A7*	Genealogy (2018)	0.151	**0.071****	0.155	0.152	0.150	0.169**	66
A8*	Stroop (2018)	0.446	**0.229****	**0.344****	**0.363****	**0.384****	**0.353****	54
A9	Train (2018)	0.196	0.160	0.168	0.187	0.196	0.176	59
A10	Genealogy (2019)	0.142	0.134	0.135	0.136	0.142	0.131	66
A11*	Stroop (2019)	0.105	**0.058****	0.112	0.107	0.104	**0.102***	75
A12*	Train (2019)	0.244	**0.176****	**0.219***	0.247	0.246	**0.204****	82
		50% time						
		CPR	APR	LPR2	LPR3	LPR4	WPR	*n*
All*		0.136	0.131	0.139	**0.132****	**0.131****	**0.128****	1307
A1	Dot Plot (a)	0.208	0.190	0.217	0.213	0.209	0.211	211
A2	Histogram (a)	0.158	0.156	0.179	0.168	0.159	0.156	177
A3*	Dot Plot (b)	0.147	0.189*	0.167**	0.148	0.147	0.147	115
A4	Histogram (b)	0.155	0.136	0.163	0.152	0.156	0.148	118
A5	Dot Plot (c)	0.151	0.159	0.158	0.150	0.151	0.147	131
A6*	Histogram (c)	0.102	0.104	0.118**	0.106	0.102	0.099	136
A7*	Genealogy (2018)	0.031	0.028	**0.022***	0.029	0.029	0.033	66
A8*	Stroop (2018)	0.239	**0.141****	**0.146****	**0.142****	**0.174****	**0.142****	67
A9*	Train (2018)	0.074	0.094	0.065	0.064	0.066	0.063	60
A10	Genealogy (2019)	0.065	0.070	0.053	0.054	0.059	0.056	65
A11*	Stroop (2019)	0.044	0.027	0.040	0.042	0.041	**0.036****	77
A12	Train (2019)	0.083	0.091	0.087	0.081	0.070	0.077	84
		75% time						
		CPR	APR	LPR2	LPR3	LPR4	WPR	*n*
All*		0.103	0.123**	0.110*	0.104	**0.101***	0.103	646
A1*	Dot Plot (a)	0.131	0.158*	0.149**	0.142*	0.134	0.138	157
A2*	Histogram (a)	0.110	0.142*	0.114	0.108	0.106	0.109	92
A3*	Dot Plot (b)	0.137	0.184*	0.150	0.135	0.137	0.141	60
A4	Histogram (b)	0.108	0.123	0.107	0.104	0.103	0.097	51
A5	Dot Plot (c)	0.120	0.122	0.134	0.124	0.121	0.127	79
A6	Histogram (c)	0.082	0.090	0.082	0.084	0.083	0.083	59
A7	Genealogy (2018)	0.018	0.016	0.015	0.017	0.016	0.019	17
A8	Stroop (2018)	0.137	0.122	0.089	0.082	0.100	0.059	16
A9*	Train (2018)	0.042	0.076	0.044	0.038	0.038	0.050	28
A10*	Genealogy (2019)	0.053	0.079	0.047	0.051	0.051	0.048	26
A11	Stroop (2019)	0.028	0.026	0.024	0.018	0.013	0.018	16
A12	Train (2019)	0.046	0.059	0.056	0.056	0.044	0.049	45

In the first column, * shows activities for which significant differences between the models were found using a repeated measure ANOVA (p-value < 0.05). In the fourth to eighth columns, * and ** show the significance of a paired t-test between the particular estimator and the CPR estimator (*: p-value < 0.05, **: p-value < 0.01).

possible. However, after the teacher started the activity, all of the students took time to read the instructions, which added time to their first subtask and made their starting slope quite different from that of the rest of the activity. This part of the activity was changed the following year, so we do not observe a similar discrepancy for Stroop (2019) (see Figure 2.10). For Stroop (2018), where there was a predictable change in time between subtasks, assuming that the progress rate could vary over time was a good choice. This result also indicates that the model performance depends on the type of activity or on the classroom. Our results do not clearly indicate which parameter for the LPR model performed best. It seems to be highly dependent on the activity and the students. The choice of the parameter is a trade-off between the noise of the measurements and the flexibility to adapt quickly to changes in progress rate.

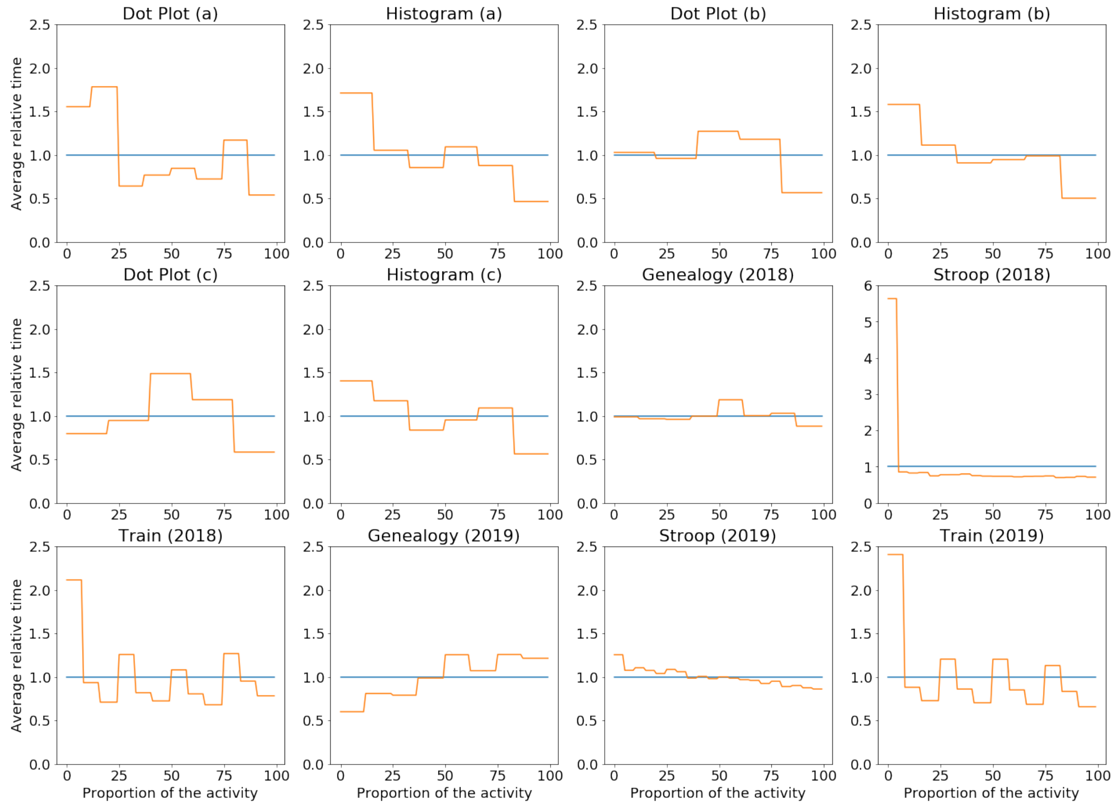


Figure 2.10 – Weight profile of the 12 activities. Because of a large divergence, the vertical scale of the weight profile for Stroop (2018) is different from that for the other activities.

Moreover, near the end of the activity at the 75% time point, there are fewer statistically significant results for the LPR models even though we often see the same trends as at earlier time points. This is most likely due to many students having finished the activity by that time point and thus being excluded from the analysis, lowering our statistical power. For example, Stroop (2018) goes from having $n = 67$ at the 50% time point to $n = 16$ at the 75% time point, which is a 76% reduction in the number of students. It may also mean that any information added from the different model variations does not apply to the last students finishing, who may perform differently than the other students.

Finally, the WPR model showed similar results to the LPR model. Like the LPR model, WPR is an adaptation of the CPR model based on the assumption that the progress proportion varies between subtasks. This model becomes very interesting in situations where the progress proportion of each subtask varies strongly from the chosen assignment. Unlike the LPR models, the WPR model takes into account varied change, where there may not be a pattern in the weights. For example, if a quiz gets steadily harder as it progresses, then there is a uniform change. However, if the difficult questions are randomly placed throughout the quiz, then the change is varied and performance on a previous question cannot be used to predict current performance. Although the weight data is still sparse at the 25% time point because few students, if any, have completed the task, it is still beneficial for predicting 9 of the 12 activities (with statistical significance for 4 activities). The WPR model may not have had a better prediction on all activities for several reasons. The first is that most of the prediction error comes from individual noise, which cannot be more accurately predicted from aggregated information about the whole class. Second, the weight profile for 11 of the activities is not as divergent from the original equal-weight assignment as for the Stroop (2018) activity (see Figure 2.10), which means that the benefits of the WPR model are not as large. Additionally, the divergence in the weight profile of Stroop (2018) is exactly at the beginning of the activity, so the WPR estimator can quickly estimate it well, but other estimators are actually more affected by this divergence. Finally, for some activities, the faster students are not similar to the rest of the class. This leads to lower prediction performance for the estimators that use aggregated data of the class (APR and WPR) because the APR and the weight model estimation will be biased toward the behaviour of the faster students.

Evaluation of Aggregated Predictions

In Section 2.4, we compared the quality of our proposed estimators for predicting individual progress. The error rate aggregated over the whole class also matters to justify the usefulness of the method for displaying class-level data. Table 2.3 shows the same error measurement as reported in the previous sections using equation (2.9), with the difference that the progress predictions are first averaged over the whole class. This leads, as expected, to lower error rates for two reasons. First, the cases where the future progress of a student is overestimated are now being compensated for by cases where the future progress has been underestimated, thus reducing the error in the aggregated prediction. The prediction error will stay high for an estimator if it mostly overestimates or mostly underestimates the future progress in a systematic way. For example, this is what happens for the APR model, which even though it has lower prediction error for each individual student at the 25% time point, it has a higher aggregated prediction error for the activities Genealogy and Histogram (a), (b), and (c). Second, lower errors occur in the aggregated data because students with progress value one, for whom predictions are trivial, were not included in the predictions in the individual case but had to be included in the aggregated case. In this case, because the students have already finished the activity, they do not have any error.

Table 2.3 – Mean Prediction Error for Aggregated Predictions at 25%, 50%, and 75% of the Time for Each Activity

		25% time						
		CPR	APR	LPR2	LPR3	LPR4	WPR	*n*
A1	Dot Plot (a)	0.140	0.213	0.146	0.141	0.140	0.126	192
A2	Histogram (a)	0.249	0.143	0.248	0.249	0.249	0.244	216
A3	Dot Plot (b)	0.152	0.242	0.153	0.152	0.152	0.151	129
A4	Histogram (b)	0.198	0.127	0.198	0.198	0.198	0.184	145
A5	Dot Plot (c)	0.096	0.210	0.093	0.096	0.096	0.094	150
A6	Histogram (c)	0.160	0.096	0.158	0.160	0.160	0.139	166
A7	Genealogy (2018)	0.085	0.047	0.086	0.086	0.085	0.086	67
A8	Stroop (2018)	0.404	0.172	0.319	0.343	0.363	0.308	54
A9	Train (2018)	0.125	0.051	0.065	0.097	0.124	0.100	59
A10	Genealogy (2019)	0.058	0.108	0.047	0.052	0.057	0.046	66
A11	Stroop (2019)	0.068	0.014	0.066	0.066	0.065	0.061	75
A12	Train (2019)	0.196	0.111	0.113	0.167	0.193	0.139	82
		50% time						
		CPR	APR	LPR2	LPR3	LPR4	WPR	*n*
A1	Dot Plot (a)	0.061	0.091	0.059	0.058	0.061	0.039	230
A2	Histogram (a)	0.032	0.053	0.037	0.034	0.032	0.026	224
A3	Dot Plot (b)	0.023	0.122	0.028	0.022	0.023	0.027	133
A4	Histogram (b)	0.042	0.024	0.037	0.040	0.042	0.026	146
A5	Dot Plot (c)	0.065	0.108	0.055	0.061	0.065	0.049	154
A6	Histogram (c)	0.008	0.027	0.036	0.021	0.011	0.023	169
A7	Genealogy (2018)	0.008	0.009	0.005	0.005	0.006	0.009	76
A8	Stroop (2018)	0.133	0.065	0.077	0.078	0.101	0.072	106
A9	Train (2018)	0.026	0.013	0.011	0.010	0.011	0.006	62
A10	Genealogy (2019)	0.030	0.047	0.013	0.018	0.022	0.012	68
A11	Stroop (2019)	0.029	0.011	0.025	0.028	0.027	0.023	84
A12	Train (2019)	0.034	0.021	0.012	0.011	0.006	0.011	84
		75% time						
		CPR	APR	LPR2	LPR3	LPR4	WPR	*n*
A1	Dot Plot (a)	0.014	0.051	0.038	0.027	0.020	0.039	240
A2	Histogram (a)	0.002	0.026	0.009	0.007	0.004	0.015	226
A3	Dot Plot (b)	0.007	0.053	0.015	0.006	0.007	0.009	133
A4	Histogram (b)	0.015	0.005	0.005	0.010	0.012	0.004	146
A5	Dot Plot (c)	0.013	0.021	0.021	0.014	0.013	0.023	157
A6	Histogram (c)	0.004	0.011	0.012	0.008	0.005	0.010	170
A7	Genealogy (2018)	0.002	0.002	0.001	0.002	0.002	0.002	78
A8	Stroop (2018)	0.017	0.007	0.010	0.010	0.013	0.004	116
A9	Train (2018)	0.002	0.019	0.007	0.004	0.004	0.004	64
A10	Genealogy (2019)	0.018	0.028	0.015	0.015	0.016	0.011	69
A11	Stroop (2019)	0.004	0.003	0.001	0.001	0.001	0.002	84
A12	Train (2019)	0.005	0.012	0.013	0.013	0.006	0.010	84

From Table 2.3, we see that the aggregated prediction errors range from 0.001 for several activities and models at the 75% time point to 0.404 for the CPR model for the Stroop (2018) activity at the 25% time point. The results seem satisfactory to be used by teachers. For example, for the 75% time mark, which is when the teacher is closer to deciding to transition to the next activity, for the baseline estimator the error is the highest for activities A8 and A10 at 1.7% and 1.8%. An error of this scale means that the prediction is off by usually less than the progress of two or three students for a classroom of about 150 students. For these two activities, we see that the WPR model has an even lower prediction error, with a decrease of 39% for A10 and 76% for A8.

2.5 Discussion

In this chapter, we aimed to address the problem of dynamically predicting students' future progress during activities in the classroom. We presented a basic model for making predictions in real time and proposed three variations of this model that accounted for different challenges of predicting the overall progress of a class through an activity. For each of our model variations, we found that we could better predict students' progress compared to the baseline model for a subset of our datasets. By analysing the differences in performance between these datasets and the time of the predictions, we can form a better understanding of completion patterns for different activity types, as well as of how students engage with these activities.

When we evaluate the patterns of the model successes as a whole, the first one that emerges is that aggregated information about the whole class can be used to improve the performance of progress predictions for individual students. It is important to note that the models using whole-class data (APR) and automated weighting of progress assignment to subtask (WPR) each increased the prediction performance for over half the activities at the beginning of an activity. Still, there are cases remaining where our results show that students' individual variability is more important than the average behaviour of the class. Thus, the prediction did not improve compared to the CPR model. However, it was uncommon for a model that used aggregated performance to perform significantly worse than the baseline model, indicating that these models could be a good starting point, independent of the activity type. Additionally, our results show us when during the in-class activity the different estimators perform best, which indicates how we could design a combined estimator by appropriately selecting the different modeling approaches that we proposed.

A second pattern that emerges is that there is less of a difference between models as time increases. As mentioned earlier, this difference could be explained by the decrease in the number of students who have not finished the activity, which leads to a decrease in statistical power for our analysis. However, it may also be due to the types of students who complete the activity later. There can be multiple reasons for students to take longer on an activity: they started later, they solve subtasks at a slower pace, or they have off-task behaviour. With the later start and the slower pace, we could expect a relationship with the way the students perform and how earlier students perform. In the case of starting later, it may just be a shift in time, while with the slower pace, it may just be extending the time frame. On the other hand, when the difference is not a simple transformation, as would be the case with off-task behaviour, the enhanced models may not improve the prediction. In future research, it may be beneficial to assess the relationship between earlier and later students' progress to investigate where these differences may arise.

Finally, we see that within any given time point, there is no clear pattern when assessing the results within an activity or class. For example, the Stroop (2018) activity performed better for all three of the LPR models, while the Stroop (2019) model did not. These results indicate that there was not a common pattern of time taken on activities across these two classes. This can be confirmed by looking at the progress curves in Figure 2.8, in which we can see that although the curves are similar, there is more of a lag at the beginning of the activity in Stroop (2018). This results in differences in the weight profiles of these two activities (see Figure 2.10). This lag illustrates how external classroom events such as an orchestration action of the teacher can affect the timing of an activity. This lag is not present in any of the other activities for the same students, so it is not a property of the students but may have been due to something that happened in the class that day. These results illustrate the importance of the timing model being able to take into account unexpected events that may happen in the classroom in real time. The proposed estimators account for these real-time events through different methods. Additionally, such events could be integrated in our estimators if they were implemented as part of the digital technology used in the classroom. How to combine our different techniques to obtain the optimal predictions will require future research.

To test our models, we used 12 datasets to see how they performed in a range of situations. Across the 12 datasets, there were 5 different activities and five different sets of students. Having an overlap of activities and students allowed us to analyse patterns that may have emerged associated with activity or student traits. As mentioned above, we found many discrepancies between the same activities and the same students when applied to different situations. These findings provide more strength to our approach, in which we do not rely on a trained model, which then may overgeneralise, and do not make any assumptions at the beginning of the activity. We are then able to apply our models to any new activity, student group, or situation without affecting the accuracy of our predictions. However, all of our datasets were collected from a higher-education setting. In this case, our results may have limited generalisability to K–12 education if students follow drastically different progression patterns.

In practice, over all the estimators we presented, the error rates for the prediction are quite satisfactory. For example, looking at the 75% time of the activities, the prediction errors in terms of RMSE are nearly never above 0.15 and are on average 0.103 for the CPR model. An error means that the prediction estimator on average wrongly predicts one more or one less subtask completed for an activity with ten subtasks. Given the unpredictability of student behaviour in general, we found this to be a very satisfactory result. Figure 2.3 illustrates well the very high variance in the time students take to complete subtasks of an activity, and Section 2.2 details the difficulty of predicting students' progress.

In addition to the error rates for the individual predictions, we also evaluated the error rate for predicting the average progress of the classroom. In this case, the errors are much lower. For example, at the time 75% (which is the most relevant time because it is closer to the moment that the teacher actually decides to transition to the next task), the error rate of the averaged prediction is always below 0.02. This could be the result of the estimator making perfectly accurate predictions for 98 students of a class of 100 students and completely wrong predictions for the remaining 2 students. This range of error rate seems likely to be beneficial for a tool aimed at supporting teachers' decisions.

The estimators described in this chapter are based solely on discrete measurements of student progress during activities that can be naturally divided into a sequence of subtasks. The estimators we presented are limited to this particular case and would not directly generalise to other contexts. This limitation could be extended in three ways. First, progress could be measured continuously and be generalised to activities that are not naturally divisible into subtasks, such as writing essays or brainstorming. A second improvement worth investigating is to extend our use of current whole-class data to improve the prediction to study how data collected previously from the same class or data collected previously from the same activity can be used in the predictions [29]. Additionally, the estimators are limited to using only data from the current activity and do not use other sources of data. This has the advantage of making the estimators easily usable without preparation but misses opportunities to increase the quality of predictions. Learning from other sources of data would also allow the use of more data-intensive algorithms for prediction, such as deep neural networks. Finally, a third extension is to use other sources of data not directly related to students' progress. For example, evaluating in real time the state of the student (idle/active) during the activity could inform the predictions to know when the student is not likely to make progress. This could be done using eye tracking or video recording [108].

Because our models allow us to predict the progress of students at a given point in time or to predict the time at which students will reach a given point of progress, they can be used in the classroom to support activity transitions. In the classroom, it can be difficult for teachers to monitor all students at the same time, particularly when students are working on individual or small-group tasks. Using teacher dashboards, teachers should be able to easily monitor and process the data and actions of students in a consolidated form [123, 206]. Our models can be used as the back end to the dashboard to provide this information to teachers. How this information is provided is still an open question, with research only beginning to investigate the different levels of orchestration support that can be provided [240]. In the case of timing, the information could be shown as a graph, as in Figure 2.5, or as a set of recommendations based on the prediction. How best to use these models to provide teachers with timely information is a direction for future research.

2.6 Conclusion

In this chapter, we presented estimators to predict the future progress of students as they work on individual activities in the classroom. Our techniques principally address the challenges arising from individual differences in students. Furthermore, two of our estimators also use data collected from the whole class to improve individual predictions. By predicting the future progress of students, we address a practical issue in the classroom of teachers not being able to fully monitor their students and not knowing when to transition to the next activity. Specifically, in large lectures, the teacher cannot move around the classroom, and common indicators of completeness, such as the noise level in the classroom, can often be misleading. By having an accurate prediction of when students may complete an activity, teachers can make informed orchestration moves, such as giving time warnings instead of suddenly switching to a new activity. Teachers being able to make more informed decisions about activity time can prevent students who have already completed the activity from waiting too long as well as preventing those who are still working on a task from getting cut off too early.

By analysing the strengths of different variations of our estimators, we gained further insight into students' behaviour during classroom activities. Our work showed how different strategies for predicting students' progress would perform based on individual differences and similarity to the average of the whole class. Overall, our work contributes to the field of learning analytics by introducing a novel approach to evaluating students' progress and progress rates during activities in classrooms. Moreover, our work has direct implications for supporting teachers' orchestration by addressing their need for time management support.

Within the context of the thesis, the work we presented in this chapter motivates the decomposition that we described in Section 1.2. Learning analytics, in particular the collection and real-time analysis of data, is a key aspect of adaptive and self-improving learning environments. The tool built for this project encompassed multiple aspects of and adaptive learning environment: data collection, prediction of students' future states, and support for optimising the learning process. Additionally, the models that we described in this chapter are relevant to Chapter 3 which focuses on simulation of students' behaviour and student models.

3 Student Models and Simulations

Observing the learning processes of students is undoubtedly an important practice to improve our understanding of learning and to design efficient learning environments. Carrying out pedagogical experiments in controlled environments is a practice that researchers use to observe students. Although such experiments can advance the learning sciences, they can be difficult to conduct because they may involve a large number of students, who are not always readily accessible. In this chapter, we specifically focus on the concept of student models and how they can be used for simulating students' behaviour or learning processes. We show that simulating students allows us to estimate statistical properties of large cohorts of students, which can inform the design of experiments on real students.

The development of Machine Learning (ML) brought valuable insights into the science of learning. Today, countless algorithms are looking at data and learning from it. Because of researchers' interest in studying the learning processes of such algorithms, the concept of Machine Teaching (MT)[1] emerged. In general, the aim of MT is to select an optimal training set of input data (or sequence of inputs) to teach a given ML algorithm a desired target set of parameters [260, 261]. Such a training set is optimal if it minimises a chosen metric, such as the number of examples or the cost of generating examples. In the case of simulated classrooms, MT can be used to optimally select a single input to teach multiple learners simultaneously [257]. That is similar to the case of a teacher that, in general, must provide a single lecture that is adapted simultaneously to all the students in the classroom.

MT has multiple goals although most are outside the scope of education. Nevertheless, part of the research on MT is motivated by optimising education. This goal relies on the similarities between the algorithms that are analysed through the MT framework and the students' learning processes [260, 261]. Thus, it is beneficial to look for algorithms that are good approximations of how students learn.

Optimising teaching necessitates being able to anticipate how students will respond to learning activities and how their knowledge will evolve throughout the teaching process. This can

[1]Not to be mistaken with Skinner's teaching machine (https://en.wikipedia.org/wiki/Teaching_machine)

be achieved using straightforward predictions but also through simulating students behaviour and performance. This distinction between the prediction and the simulation of students' behaviour is a concept of ML, which differentiates between discriminative and generative models [165].

In Section 3.1, we review uses of student simulations within education. In Section 3.2 we focus on the distinction between discriminative and generative models. In particular, we look closely at Bayesian Models and the example of Bayesian Knowledge Tracing (BKT) because they are the most straightforward application of the framework that we described in the introductory chapter. In Section 3.3, we report on our study using Semi-Markov models (SMM) for simulated MOOC students. This study proposes a general method for the simulation of users' activity that can be used in a wide range of applications. Finally, we argue that simulations can also be used to prepare for large scale experimentation on learners. This is further justified by our discussions in Chapter 4 on algorithms for self-improving learning environments.

3.1 Student Simulations

Simulating students' behaviour lies at the intersection of cognitive science, artificial intelligence, and education. Researchers have studied methods for the simulation of human behaviour outside the field of education [26] and notably in a web context [42]. An example of the usage of simulating humans (not students) for education deals with simulations of patient behaviour for training medical students [91].

Educational researchers have uses simulations of students with multiple goals. One goal is to help teachers practice instructional skills. An example of this goal is to simulate students' errors in order to train teachers to recognise students' misconceptions [246]. Student simulations have also been used to help author tutoring systems by having the teacher demonstrate the desired tutoring strategies to the simulated students [147]. The tutor is, in that context, automatically built from the demonstrations of the teacher and, thus, does not require programming expertise. Simulated students can also support the automatic evaluation of Intelligent Tutoring Systems (ITS) because statistical inference can be performed efficiently on the output of a simulation while other methods for evaluating ITS can be computationally intractable [90]. Additionally, doctoral programs have been simulated as a way to analyse different designs without requiring tests with learners, which would require a much longer time [131]. Finally, another encouraging use of simulated students is the possibility of using such simulations of students for the pedagogical practice of learning by teaching [21, 22]. For example, by using a robot that simulates bad handwriting and asking a child to help the robot improve its handwriting, the child can learn to write better by him or herself [105].

Because student simulations can have multiple goals, several methods can be used for the evaluation of simulation models [124]. For example, in the case of simulations used for learning by teaching or authoring ITS, the simulation can be evaluated from the resulting outcome. In other cases, a simulation could be evaluated, either by analysing the properties of the underlying cognitive architecture or by testing whether optimal teaching for the simulated learners is also optimal for human learners. In this chapter, we design a simulation model that seeks to describe and reproduce MOOC students' behaviour. To evaluate the quality of our simulations we compare statistical properties of the simulated and real data.

3.2 Algorithms for Student Simulations

Modeling students is a key concept in learning analytics and educational research in general [211]. It is also a required component of building students' simulations. The vast amounts of data that can be gathered and analysed in modern learning environments allow us to build models of unprecedented scale and accuracy. For example, researchers have built such models for predicting motivation and cognition [254], or for predicting student's goals [75]. Large datasets allow researchers to find predictive power of seemingly slightly related signals or potentially noisy signals. For example, the length of pauses in a MOOC video estimates the difficulty perceived by students [132] and the head movement of students estimates the level of attention during lectures [188]. All the aforementioned models are focused on prediction and belong to the class of so-called discriminative models. Although these models can be used to simulate students by generating predictions, they are less suited to the task of simulation as generative models.

3.2.1 Generative models

A common area of ML is the problem of supervised learning [201]. Supervised learning consists of learning a mapping between inputs (x) and outputs (y) based on the observation of several of these (x, y)-pairs. The task to be solved is to find the new value of y given a new value for x. In the case of education, an example could be that the input x is a vector containing several features describing the behaviour of a MOOC student, Alice, in the first week of the course and y is the final score at the end of the MOOC [110].

A discriminative model is a model that will focus solely on the prediction of the output y by learning the probability distribution $P(y|x)$ (probability of y given x). This probability distribution allows us to answer the question in our example: "Alice had behaviour x during the first week, what final score y will she have at the end of the MOOC?". On the other hand, a generative model will learn the joint probability distribution $P(x, y)$. From this joint probability distribution it is possible to compute the distribution $P(y|x)$ used by discriminative models, but it is also possible to compute the distribution $P(x|y)$ (probability of x given y). This second distribution allows us to answer the question: "Alice had a final score of y, what behaviour x did she have during the first week?". A generative model, not only contains information about the relation between x and y, but also about the marginal distributions of x and y. Thus, unlike discriminative models, generative models are able to describe what typical students are like.

Generative models allow one to simulate students by generating samples from the $P(x, y)$ distribution, which in our example tells us how likely each type of behaviour and final performance is and how they are dependent on one another. In particular, using a purely discriminative model for predicting performance in a MOOC would not allow one to give estimates for the types of behaviour expected from students during the first week of the course. That is a strong limitation if the model is to be used to simulate students' behaviours and learning processes. Generative models, however, capture explicitly the underlying process.

In the study detailed in the following Section 3.3, instead of focusing on prediction, we model behaviour itself in a probabilistic manner. This approach can not only significantly improve already established techniques but also provides a basis for simulating students' behaviour. We use a generative model, which allows us not only to predict, but to generate observations from the estimated distribution of students in a MOOC. These models capture both the probability structure of input variables and the flow of the processes.

Generative models in education serve mainly for understanding, visualising and predicting students' behaviour. For example, analysing MOOC forum's messages using a generative model allowed to classify discussion thread efficiently and to rank threads by relevance [30]. Among the many generative models, Markov models are used for visualisation [49], for modeling engagement [193] and for modeling students retention [16]. Apart from Markov models, we encounter models such as mixture models and Bayesian models [212] and BKT in the context of MOOCs [145, 173].

3.2.2 Bayesian Knowledge Tracing

BKT [55] is one of the most popular models for real-time knowledge tracing. The general goal of Knowledge Tracing algorithms is to provide estimates of students' knowledge over time as students are interacting with pedagogical material (most likely in an ITS). In the case of BKT, this estimation is a probability of skill mastery. It is to date still one of the simplest and most used Bayesian models.

Figure 3.1 gives a graphical representation of the BKT model. It reveals several underlying assumptions:

- **Binary representation of student's Knowledge States**: For a particular skill (sometimes called a Knowledge Component), the student can either master it or not.
- **Probabilities of observations**: A student who masters a given skill is expected to answer questions and problems about that skill correctly, but it can happen with probability P_S that the student makes a mistake ("slip"). Similarly a student who does not master a skill could still answer a question correctly out of luck ("guess") with probability P_G.
- **Probability of initial mastery**: Some proportion P_0 of the students will have already mastered the skill before their first interaction with it in this session.

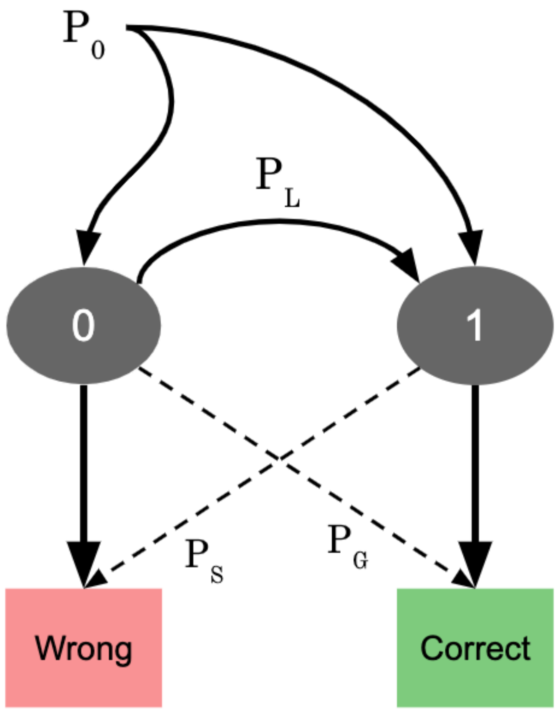


Figure 3.1 – Graphical Representation of the BKT model

- **Probability of transition**: With a fixed probability P_L, students transition from non-mastery to mastery. Once a skill is mastered it is never forgotten.

Although some of these assumptions might seem too simplistic to be accurate, they are nevertheless useful due to their high predictive power [89].

Variations of BKT

Bayesian models can be very expressive, but BKT has limiting assumptions of BKT, researchers have developed several variations of the model. These variations of the BKT model which discard one or several of these assumptions have been developed. Typically, the lack of forgetting could be considered one of the main drawbacks of BKT. The multiple variations of BKT show the wide range of behaviours that can be expressed with Bayesian models.

A first variation improved the models' predictions by using individualised parameters for the probability of initial mastery and the learning rates [174, 258]. A second variation models the relationships between skills as a Bayesian network, which outperformed BKT in prediction accuracy [117]. The assumption of binary correctness has been removed with a model that allows for continuous scores [249]. The assumption of constant guessing and slipping rates have been removed by considering the difficulty of different exercises [175]. Additionally, another approach introduced the possibility of forgetting a mastered skill [187].

3.2.3 Bayesian Models

Many of the variations and improvements made on BKT remain in the general category of Bayesian Models. Bayesian Models are a more general framework to model how a system evolves over a sequence of steps, while possibly generating observations and being acted upon. This type of models has been used extensively in multiple educational contexts [116, 157]. For example, they have been used for adaptive testing and diagnostics [158], generating hints in ITS [86] and inferring students' goals and attitude during problem-solving activities [9, 52]. Another aspect which makes Bayesian Models particularly useful is their ability to evaluate the uncertainty about the state of the students [51]. Additionally, Bayesian Models (other than BKT) have been used to compute fast and adaptive teaching strategies [190] and to optimise ITS using decision theory [153]. Millán, Loboda, & Pérez-de-la-Cruz [156] consider Bayesian Models "one of the best options for building student models." We fully concur with this statement because Bayesian models are one of the most straightforward applications of the framework described in Section 1.2. A Bayesian Model consists of several components that reflect on our proposed framework:

- A set of the possible states of the student: $s \in \mathscr{S}$
- A distribution of probability of observations: $o \in \mathscr{O}$, and $P(o|s)$
- Transition probabilities between states $P(s'|s)$, the probability for the student to be in state s' after being in state s

To be useful in adaptive learning environments, the model should also include a set of actions $a \in \mathscr{A}$. In the case of a learning environment, the actions can be different learning activities to give to students, different exercises, or different pedagogical interventions such as giving a hint or feedback. In the general case, the actions taken by an adaptive learning environment will influence the probabilities of observations and the probabilities of transitions. For example, we can consider an adaptive learning environment that can choose between providing either a quiz or a video. If a quiz is given, the observations will be the student's answer to the quiz. If a video is given, the system will be able to observe the students' interactions with the video player, such as pausing or speeding up the video [133]. It is also expected that the two types of learning activities will affect the state of the student differently. How actions taken by the learning environment influence the learning process and how to correctly select actions have been studied using Partially Observable Markov Decision Processes [190] (which are identical to the above definition).

Using Bayesian Models, the state of the student is estimated again after every observation using Bayesian inference. At time step t, the model has uncertainty about the state of the student. This uncertainty is characterised by a probability distribution $P(s)$ for $s \in \mathscr{S}$. Then after choosing action a and making the observation o, the new probability distribution about the new states s' can be computed following the rule in equation 3.1. It is important to note that for very complex student models involving a large number of states, actions and observations and without simplifying assumption, this estimation will be computationally intractable in the general case.

$$P(s'|s,a,o) = \frac{P(o|s',s,a)P(s'|s,a)}{P(o|s,a)} \tag{3.1}$$

3.2.4 Other Knowledge Tracing Models

Other Knowledge tracing methods are based on different modeling approaches, such as Learning Factor Analysis [38] or Performance Factor Analysis [176]. These two models rely on logistic models based on the difficulty of exercises, the ability of the students and the sequences of successes and failures in previous exercises. A more recent and very promising modeling approach is to use Deep Learning methods such as Long Short Term Memory (LSTM) [150, 180] or Transformers [186].

3.3 Study: Semi-Markov Models for Simulating MOOC Students

This section contains the methods and results of a study that aimed to develop a practical algorithm for simulating students behaviour in MOOCs [77]. We analysed sequences of activities from 61 courses using SMM and the Expectation-Maximisation algorithm (see Section 3.3.2 for detailed descriptions of the algorithms). The simulations consider four types of learning activities: watching a video, submitting an assignment, visiting the forum, and posting a message on it. We used SMM because we found them to be more efficient to not only simulate the transitions between events but also the number of repetitions of each event. This efficiency is because it models different probability distributions for the repetition and the transitions between events. We used Expectation-Maximisation (EM) to compute the parameters of the Bayesian models as well as to cluster the students according to their different types of behaviour and to compute the transition probabilities for each cluster. We balanced the complexity of the model's structure and the number of parameters by cross-validating its parameters. Finally, we used the computed models to simulate students activities and examine how well the simulation reflects student's behaviour, which allowed us to predict events of other courses compared to straightforward techniques.

The study aims to answer three research questions: **RQ1:** To what extent can Semi-Markov chains be used to describe behavioural patterns of students? **RQ2:** How interpretable are clusters of students based on statistical differences in the students' behaviour? **RQ3:** How can these models be used to infer distributions of events?

Abbreviation	Description	Proportion
`VideoPlay`	watching a video	51%
`Submission`	submitting an assignment	33%
`ForumView`	visiting the forum	15%
`ForumPost`	posting on the forum	1%

Table 3.1 – Distribution of events in the dataset.

3.3.1 Methodology

Dataset

From our internal MOOC database, aggregating data from Coursera and edX, we extracted events for 61 EPFL courses. The raw data contained approximately 23 million events for 500,000 students, arranged in tuples: `<StudentID, CourseID, EventType, Timestamp>`. The *EventType* describes the type of activity and takes one of four possible values presented in Table 3.1. Note that our modeling technique can be easily extended to cover other types of events.

Analysis

For the analysis we developed our own Python implementation of the algorithm fitting the model[2]. Below we explain the algorithm in detail. Since 23 million events can still fit in memory of a single computer, we did not require a specific computing architecture to perform the analysis. However, given the considerable size of the dataset, the algorithm takes several minutes to run.

Probabilistic Model

We start with a general model in which students' activity in any MOOC can be very precisely described. Next, we elevate abstraction of the model by adding assumptions simplifying the analysis. Our goal is to introduce a model whose complexity can be adapted to a particular course and the amount of available data. We consider a model in which students' behaviour is described in a sequential manner by the type of activity they perform and the time they wait between two sessions. Furthermore, as most of the students perform at most one MOOC session per day, we choose a daily granularity of actions.

A sequence of student's daily activity is described as a list of 'active events' (`VideoPlay`, `Submission`, `ForumView` and `ForumPost`) followed by a 'end of the day event' (`EndOfDay`) or only a `EndOfDay` in the case the student did not perform any activity the given day. For example, if a student plays two videos on Monday and then posts a message on the forum on Thursday, his sequence of events for these days takes form:

[2]Our implementation is available under https://github.com/lfaucon/edm2016-mooc-simulator

```
Mon. VideoPlay/VideoPlay/EndOfDay
Tue. EndOfDay
Wed. EndOfDay
Thu. ForumPost/EndOfDay
```

The formal definition of the model is:

- **The set of all students** S**:** We use the symbol $s \in \mathscr{S}$ to designate an individual student.
- **The set** $\mathscr{A}$ **of all types of activities:** For this study we chose a set of four types of events presented in Table 3.1 as well as the special event, `EndOfDay`. We use the symbol $a \in A$ to designate any type of activity. One can extend the set of activities to other events if needed for a certain application of our model.
- **The random sequential variable:** $X_1^{(s)}, X_2^{(s)}, \ldots, X_n^{(s)}$ represents the sequence of activities of one student s. Each $X_i^{(s)} \in \mathscr{A}$ and the sequence stops after an `EndOfDay` when the student reaches the end of the course. We denote the length of the sequence for a student s as $n^{(s)}$. The observation of one student activity along one MOOC is a **realisation** of the random sequence X.
- **The probability distribution** P**:** In general, for each student $s \in S$ we can model the i-th event $X_i^{(s)}$ with a probability distribution

  $$P^{(s)}(X_i^{(s)} = a \mid X_{i-1}^{(s)},, X_{i-2}^{(s)}, \ldots, X_1^{(s)}, C_s),$$

  where $a \in A$, $X_1^{(s)}, \ldots, X_{i-1}^{(s)}$ are the previous events of that student and C_s are individual characteristics of the student.

This distribution represents the student's behaviour profile and allows us to generate typical sequences of activities. Our main objective is to model this distribution as accurately as possible, given the limited information. The accurate distribution would allow us to draw samples of simulated student data. These probabilities correspond to the requirements for classifying and simulating student behaviours.

At this point, one student interaction with a MOOC can always be represented as a sequence of activities taken from the set A. Note that we do not specify the regular 'end of a course' event, since we only model the behaviour within the limited time-frame of a course and we treat the last day of the course as the last day of the process. Therefore, each student who went through the whole course without dropping out has just a `EndOfDay` event on the last day of the course. Therefore, the number of `EndOfDay` events is equal to the number of days of the course. Ideally, we are able to simulate a student's sequence of learning activities from the probability distribution of the student. However, even in a large MOOC dataset we cannot afford estimation of the probability depending on the whole sequence, since even with just 20 steps we have $4^{20} \sim 10^{12}$ possible sequences of activities which makes estimation infeasible in practice. Moreover, we do not have any individual characteristics C_s of a student.

Simplifying Assumptions

In order to fit a probabilistic model we need to relax the dependencies on too many events in the past and on individual characteristics of students. We introduce the following assumptions:

- **A1**: Students' behaviours fit into a small number of natural categories of behaviour.
- **A2**: The type of activity depends only on the previous activity and not on old past activities.

Assumption **A1** maps the space of all possible students' characteristics into a limited number of categories. Allowing only a small number of possible categories of students, simplifies a lot the estimation of students characteristics that the algorithm must work with. Many studies on MOOCs explicitly classify students into a small number of categories [122]. Students are divided between "Viewers" who only watch videos, "Forum Actives" who share with their peers in the MOOC discussion forum and "Completers" who succeed in the assignments. As we present in the next section, our method is based on unsupervised clustering, where groups emerge in the way optimal in terms of maximum likelihood of the model.

Assumption **A2** imposes that only the last activity has an impact on the current activity. This assumption is more constraining, but since the complexity of history grows exponentially with the number of steps, and, in order to be able to estimate parameters, we have to reduce the search space. This simplification is usually called the Markov assumption and greatly reduces the search space of parameters, which makes the computation of the model feasible. Even though the behaviour and learning process of students could have long term dependencies as explained in the previous subsection, in our case the simplifying assumption is necessary to reduce the complexity of the model and make it tractable. Additionally, this simplification enables interpretations of the models as we explain in Section 3.3.3.

Apart from technical assumptions required for Markov Models, we impose other assumptions for convenience. First, we consider only the five types of events mentioned previously although other sources of data about students would be available in most MOOCs (for example, the click-stream interaction with the lecture videos, or the text of messages posted on the forum). Second, we do not consider the duration of events, so the `VideoPlay` event is only the moment when a student starts watching a video. Third, if the series of events happens during midnight, we still add an `EndOfDay` event to the sequence.

3.3.2 Algorithm

Soft clustering

In Section 3.3.1, we proposed a simplified model in which we assume that there are only a few different possible classes of students **(A1)**. We enumerate clusters $1, 2, \ldots, K$. For each student $s \in S$ we introduce a probability distribution $\mu_k^{(s)}$ that describes probability that the student belongs to the behaviour classes k, for $k \in \{1, 2, \ldots, K\}$. Furthermore, every student of the same profile behaves according to the same probabilities.

This technique is often referred to as *soft clustering, weighted clustering* or *fuzzy clustering* [166]. Instead of discrete cluster assignment, as in K-means, we obtain for each student a probability distribution among the clusters. These probabilities can be intuitively seen as our certainty that the student belongs to a given cluster.

Semi-Markov Chain

Assumption **(A2)** (i.e. dependence only on the last state) allows us to model the process Markov Chains. Formally, in the definition of distribution of the next event we can drop dependence of the events that occurred before the current one. This results in the simplification given in Equation 3.2.

$$P^{(s)}(X_i^{(s)} \mid X_1^{(s)}, \ldots, X_{i-1}^{(s)}) = P^{(s)}(X_i^{(s)} \mid X_{i-1}^{(s)}) \tag{3.2}$$

A preliminary analysis revealed an important weakness of using classic Markov Models in our context. A traditional Markov model considers that a student is equally likely to stop watching videos when they have watched one, as when they have already watched ten videos. In practice, students watch videos sequentially and a Markov Model does not capture appropriately the number of events in the sequence. To remedy this issue we employed Semi-Markov Models (also called Markov Renewal Processes). The key feature of this model is that it allows one to replace the self-loops (transitions from one event type to itself) in the Markov Chain, by a probability distribution of the number of repetition of a given state.

In Semi-Markov Models, we still need to choose a parametric distribution, but we have more freedom than in a traditional Markov Chain. A Markov Chain implicitly assumes that the number of repetitions of a given state follows a geometric distribution. As a consequence, the probability of staying in the same state is the largest for 1 step and decreases with the number of steps. However, we would expect that 1 is not the most probable number of repetitions, at least for a particular group of students. Suppose we expect that some students connect to a MOOC twice a week, with an approximately three-day interval between connections. In that case, the average number of repetitions of the `EndOfDay` event is 3. A traditional Markov Chain accurately models the average to be 3 but implicitly assumes that the most likely number of repetitions is 1. A SMM with a Poisson distribution also gives the average equal to 3, but the distribution is concentrated around 3. Using a Poisson distribution proved to be more accurate in our preliminary analysis. Thus, for an event $a \in A$ and a class k, we model the number of repeated events R_a^k by Equation 3.3 where r is the number of repetitions and λ_a^k is the average number of repetitions that needs to be estimated from the data for each k and a.

$$P(R_a^k = r) = \frac{e^{-\lambda_a^k}(\lambda_a^k)^r}{r!} \tag{3.3}$$

Expectation-Maximisation

The Expectation-Maximisation (EM) algorithm is an iterative technique used to compute the parameters that maximise the likelihood of a given probabilistic model [63]. The EM algorithm has been proven to converge at least to a local minimum. This minimum depends on the initialisation point. Thus, multiple runs with different random initialisations are often used in practice in order to increase the chances of finding the global minimum.

In this study, we use the EM algorithm for unsupervised learning. Neither the parameters of the latent classes nor the repartition of the students are known at the beginning and the algorithm has to estimate both quantities at once. In our settings, we define for each $k \in \{1, 2, \dots, K\}$ and states a and b:

- $p_{b->a}^{(k)}$, the probability that a student with the behaviour profile k performs the activity a after the activity b: $p_{b->a}^{(k)} = P(X_i = a \mid X_{i-1} = b)$
- $\lambda_a^{(k)}$, the average number of repetitions of an event a from a student of profile k.
- $\mu_k^{(s)}$, the probability that a student s belongs to the profile k.

Using these parameters, we can compute the likelihood of the observed sequence as a function of cluster repartition and parameters of Markov Chains. For this process we use the following Equation 3.4.

$$likelihood = \prod_{s \in S} [\sum_{k=1}^{K} \mu_k^{(s)} \prod_{(a,b,r) \in T_s} p_{b->a}^{(k)} P_{\lambda_a^{(k)}}(r)], \tag{3.4}$$

where T_s is the set of tuples $(a, b, r) \in A \times A \times N$ corresponding to transitions from activity b to activity a with r repetitions of activity a. The goal of the algorithm is to find the parameters that maximise the likelihood.

In the first stage, the algorithm initialises randomly K profiles. Next, it iteratively improves the *likelihood*, by alternating two steps as described below. In each step it modifies the repartition of the Markov chain parameters.

Initialisation: The initialisation consists of choosing randomly either the $p^{(k)}_{b->a}$ and $\lambda^{(k)}_a$ or the $\mu^{(s)}_k$. In our algorithm, we start with the $\mu^{(s)}_k$. This can be done by generating a random number k^* from 1 to K for each student s and by setting

$$\mu^{(s)}_k = \begin{cases} 1 & \text{if } k = k^* \\ 0 & \text{otherwise.} \end{cases} \tag{3.5}$$

Iterations: The iteration phase has two steps. First, we compute the optimal values for $p^{(k)}_{b->a}$ and $\lambda^{(k)}_a$ given that $\mu^{(s)}_k$ are fixed (equations (3.6) and (3.7)).

$$p^{(k)}_{b->a} = \frac{\sum_{s\in S}\sum_{(a,b,\_)\in T_s} \mu^{(s)}_k}{\sum_{s\in S}\sum_{(\_,b,\_)\in T_s} \mu^{(s)}_k} \tag{3.6}$$

$$\lambda^{(k)}_a = \frac{\sum_{s\in S}\sum_{(a,\_,r)\in T_s} r\mu^{(s)}_k}{\sum_{s\in S}\sum_{(a,\_,\_)\in T_s} \mu^{(s)}_k} \tag{3.7}$$

Next, we compute the new values of $\mu^{(s)}_k$ according to the new $p^{(k)}_{b->a}$ and $\lambda^{(k)}_a$ (equations (3.8)).

$$\mu^{(s)}_k = \frac{\prod_{(a,b,r)\in T_s} p^{(k)}_{b->a} P_{\lambda^{(k)}_a}(r)}{\sum_{c=1}^{K}\prod_{(a,b,r)\in T_s} p^{(c)}_{b->a} P_{\lambda^{(c)}_a}(r)} \tag{3.8}$$

Intuitively in the first step, we compute the parameters of the latent classes given the repartition of the students, and in the second step, we recompute the repartition from the new classes parameters.

The algorithm takes as input the dataset of the students' sequences of activities and a desired number of clusters K and outputs K Semi-Markov models. Each Semi-Markov model is described by transition probabilities and the average number of repetitions that represent K categories of behavioural patterns. Along with these models, for each student the algorithm returns the posterior probability distribution of the student among the clusters.

3.3.3 Results

Studying these behavioural patterns provides insights on the types of behaviour that MOOC students can have. In order to present the process of the analysis and explain the results, we first execute the algorithm with $K = 3$ clusters. As an example of output result, Figure 3.2 shows the graphical representations of 3 Semi-Markov models that correspond to one run of the algorithm. The thickness of the arrows corresponds to transition probabilities and the color of states corresponds to the average number of occurrences. We provide this example to illustrate the function of the model and to use it as a basis for interpretation.

The parameter K, the number of clusters, can be set to different values depending on the objective. For higher interpretability, smaller values of K are preferred, since a greater number may result in classes having only subtle, indistinguishable differences. On the other hand, a greater number of clusters can be used for more advanced analyses when we study the generative models by comparing the results of a given algorithm using them. The common method to decide the right parameter K is cross-validation. Note that the parameter controls the complexity of the model, and high numbers can lead to overfitting if not enough data is provided.

Example: Interpretation of clusters (K=3)

Our second research question relates to the interpretability of the cluster of students that we infer from statistical patterns of sequences of events. We demonstrate the interpretability of our simulation model by showing how easily it can be interpreted. To that end, we illustrate the behaviour of the algorithm and the model when the number of clusters is small ($K = 3$). Although we may lose important variability among groups of students in this case, a small number of clusters allows us to visualise the Semi-Markov models and interpret each of the clusters.

The visualisations of the Semi-Markov models on Figure 3.2 reveals general characteristics of students' behaviours. For example, Profiles 1 and 3 are less active as they have higher transition probabilities and a higher expected number of repetitions for the `EndOfDay` events. Profile 2 has a high number of repetitions for the `ForumView` events and very balanced probability transitions between all types of events. Students adopting Profile 2 are students who use all the different learning activities of the MOOC. On the contrary, Profile 3 has a very high average number of repetitions on `VideoPlay` and considerable probability to go back to `EndOfDay` events. This means that students of this cluster are not fully engaged in all MOOC activities and mostly log in to only watch the videos.

As mentioned in Section 3.2 of this chapter, the model we used is in the family of generative models. It allows us to sample sequences of simulated student data. An insightful way to analyse and interpret the differences between the extracted clusters is to generate sequences of events and compare statistical properties of the outcomes. For example, we can compute

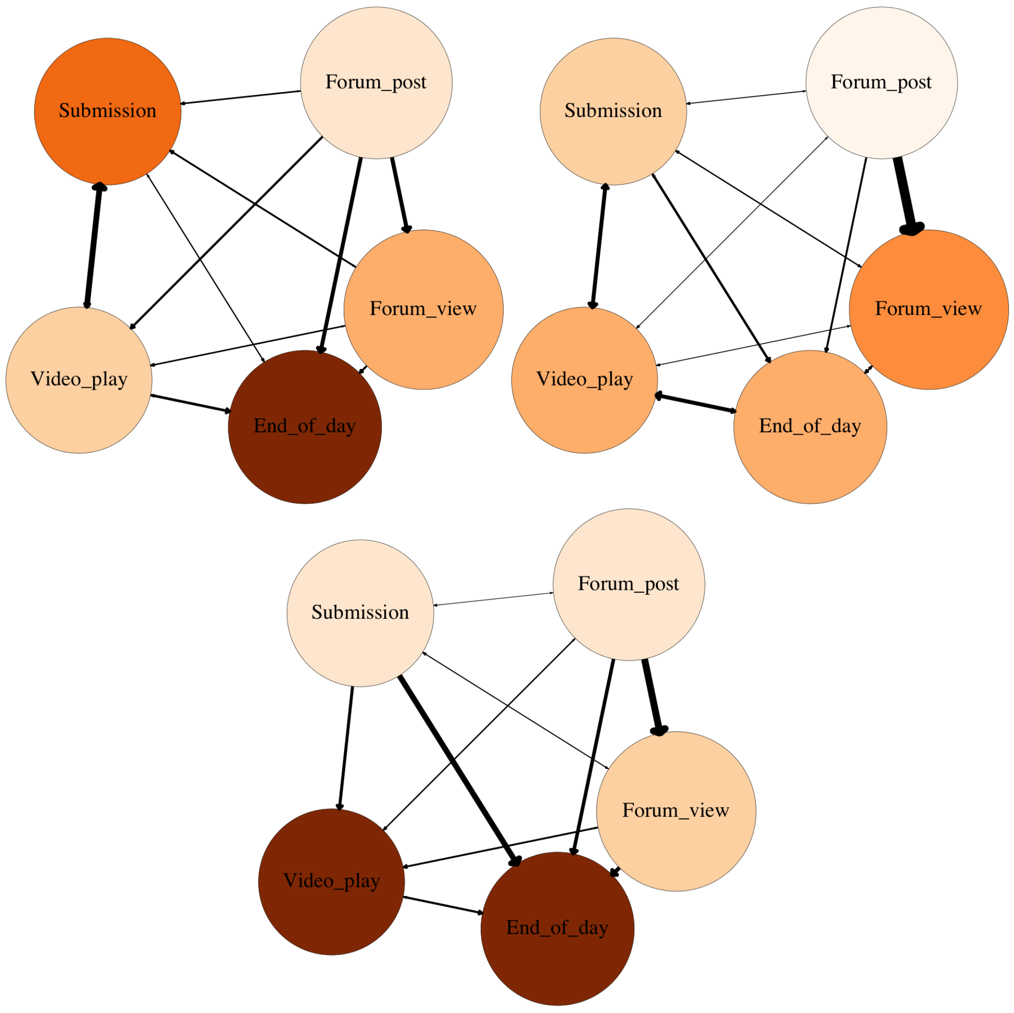


Figure 3.2 – Three graphical representations of behaviour profiles extracted by the EM algorithm. The three Semi-Markov models are referred below as Profile 1 (top-left), Profile 2 (top-right), and Profile 3 (bottom). The thickness of arrows shows the transition probability and the color shows average number of repetitions (from 1 repetition [white], to 9 repetitions [darkest brown])

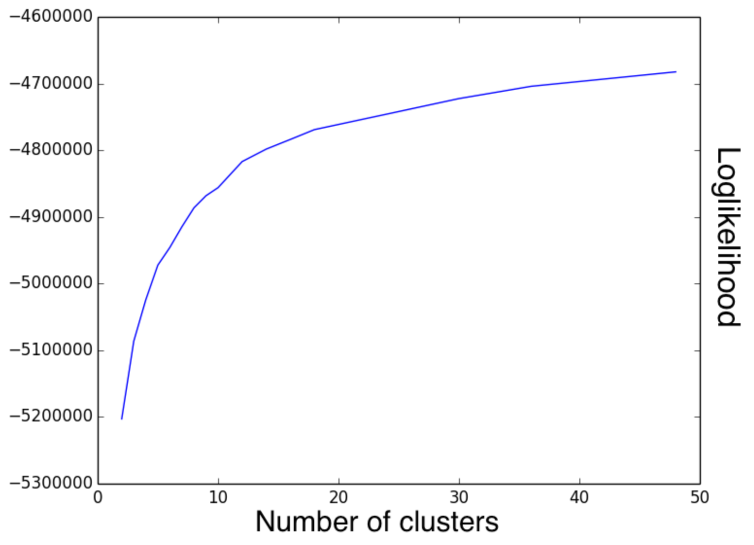


Figure 3.3 – Evolution of the log of the likelihood for different number of classes

the expected number of videos watched or the expected number of posts on the forum directly from simulated sequences. Table 2 shows the average number of several types of events for 100 simulated students (averaged from 10000 simulations) over four weeks generated with the three Semi-Markov models from Figure 3.2. For example, we can see that students of Profile 1 participate in the discussion forum more rarely but complete the assignments more than they watch the videos. This might indicate that they already have a good understanding of the content of the course and do not need to spend more time studying.

Profiles	1	2	3
Watched Videos	1060	3133	2363
Submissions	1535	2423	442
Forum Visits	68	1711	255
Forum posts	3	96	15

Table 3.2 – Average number of events for 100 students over the first four weeks of the MOOC

Choice of the parameter K

A common challenge of unsupervised learning and fitting a probabilistic model is finding the correct number of classes. In our case, the similarity of the algorithm with other clustering techniques, such as the K-means algorithm, leads to the "elbow heuristic", often used in practice. The idea is to choose the number of clusters large enough to explain a large part of the variability but such that a greater number of clusters would not explain substantially more.

Figure 3.3 shows the evolution of the log-likelihood computed by the algorithm after convergence with the equation (3.4). We can see an elbow shape for a number of clusters between 10

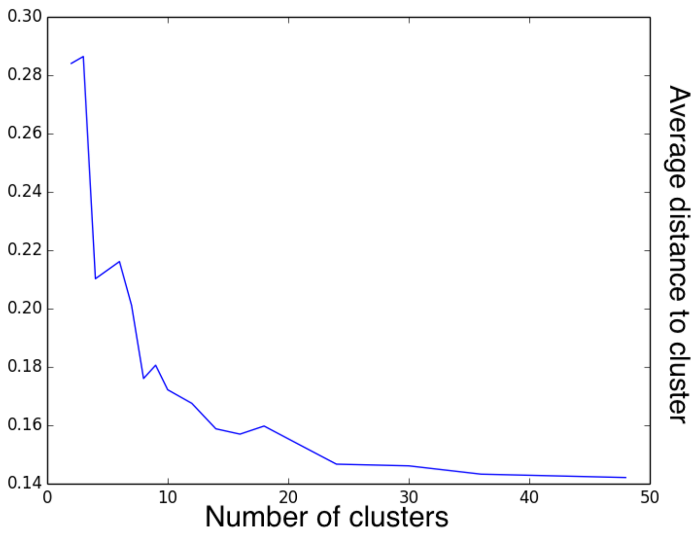


Figure 3.4 – Average distance of students from their model for different number of classes

and 15.

In order to confirm the result of this first measure of quality, we designed another measure described in the equation (3.9). The goal is to quantify how the student sequences diverge from their attributed cluster. In the equation, $|A|$ is the cardinality of the set of possible activities. $p_s(a)$ is the probability of finding the activity a if we take uniformly at random an activity of student s. $p_k(a)$ is the probability of finding the activity a if we take uniformly at random an activity from a sequence generated by the class k.

$$d^2(s,k) = \frac{1}{|A|}\sum_{a\in A}(p_s(a) - p_k(a))^2 \tag{3.9}$$

This distance measure shows an elbow shape for the same values of K between 10 and 15 as can be seen on Figure 3.4. We conclude that MOOC students from our dataset can be meaningfully clustered into 10 – 15 different classes.

Simulations

One of our motivations for designing generative models was their ability to simulate sequences of events. This relates to our third research question about the estimation of distributions of events and behaviours. These estimations can benefit the design of an experiment by decreasing the risk of having too large or too small samples because the simulation of students behavior will allow us to estimate in advance the sample sizes necessary to obtain the desired statistical power on particular conditions such as forum use or videos watched. In this section, we show how to use the generative models to simulate students' event sequences, and we compare the simulations to real students activities.

With a model fitted with the EM algorithm at hand, the algorithm partitioned students and chose parameters of a Semi-Markov Chain for each of the clusters. Since both the repartition and the Semi-Markov Chains are generative, we can draw samples from the fitted distribution. In other words, we can simulate the students. To validate the quality of the simulations, we first propose a simple accuracy measure. We use the Mean Square Error (MSE) as defined in Equation 3.10 because it is a useful metric for estimating the errors of probabilistic predictions. $P_{real}(|a| > n)$ represents the probability that a student performs more than n events of type a during the time of the MOOC. $|a|$ is the count of events of type a. $P_{sim}(|a| > n)$ represents the same probability but for a simulated student. In the measure, we chose the value $N = 50$ because it covers most of the variability in the students activity sequences and is not too large as still 19% of the students have an activity with more than 50 repetitions.

$$MSE = \frac{1}{(|A|-1) * N} \sum_{a \in A} \sum_{n<N} (P_{real}(|a| > n) - P_{sim}(|a| > n))^2 \tag{3.10}$$

In order to prove the correctness of the modeling method, we divided our dataset into a training set and a test set for validating the results. The first step is to run the algorithm on the training set with several values of the parameter K and then, use the computed parameters to simulate a new population of students and finally compare this population with the students from the testing set. In Figure 3.5 we can see that the fit does not improve much after $K = 15$ because a too high number of clusters makes the algorithm learn mostly the noise from the random actions of the students instead of their real intrinsic behavioural patterns. It appears that we have an elbow shape for K between 10 and 15, which confirms the previous conclusions.

The small error proves that the distribution obtained from simulations is close to the original distribution. This implies that the model, when properly trained, can be extrapolated by simulation to further events or larger samples. In an experimental setup, simulations with varying initial conditions of the model (e.g. probabilities of transitions) can give us distributions of events at the later state. Given multiple experimental conditions, knowing the probability distributions of the results of the different conditions allows us to estimate sample sizes needed for finding statistical evidence of an investigated effect. For further technical details on how to choose the sample size for experiments based on distributions, we refer to [142].

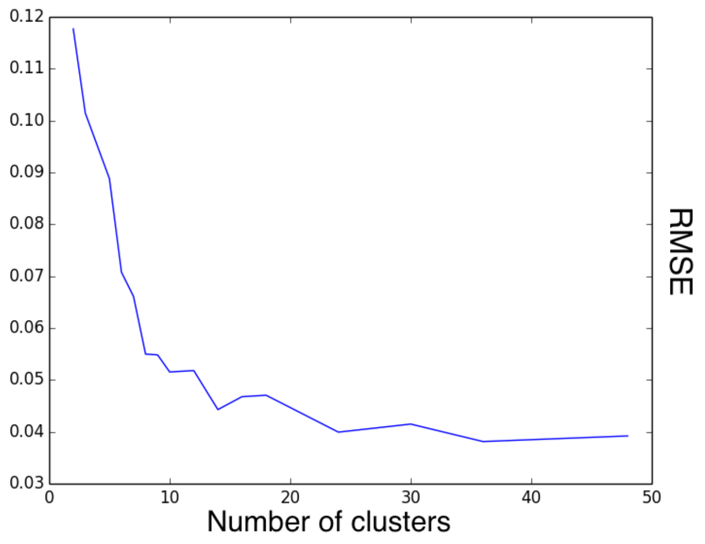


Figure 3.5 – Measure of accuracy of a simulation for different number of classes

Prediction of the cluster

Once we run our algorithm with a desired number of clusters K, we obtain the desired repartition and K corresponding Markov models, representing the extracted behaviour profiles. A natural question arises, whether we can predict the behaviour cluster for a new student. This question could be significant for the MOOC platform, as detecting the type of student that is taking a course allows the teacher to adapt the interaction that the platform offers. For example, if we are sure that a student is only going to watch videos, we could stop interrupting the videos with quizzes and sending email about the discussions on the forum. Alternatively, we could trigger an action that would aim to motivate the student to engage in quizzes and discussions. On the other hand, a student that shows the behaviour of strong involvement in completing the full course and collaborating on the forum may appreciate more external news and challenges from the course.

To investigate how well we can predict the behaviour cluster of users, we divided our dataset into a training set of 40 randomly chosen courses and a testing set of 10 courses. First, we use the selected training set to learn 3 different types of behaviours. Then we try to predict for each student of the testing set to which cluster he belongs, using only a limited number of his activities. This allows us to justify that the method can be applied to new or currently running courses.

To compute the behavioural pattern of a student, we first estimate the probability that he belongs to any cluster k using equation 3.8, and predict the most likely cluster. Figure 3.6 shows the performance of our prediction when it has data from a limited number of weeks as input. We measure the performance using the average of the F1-measure for each cluster.

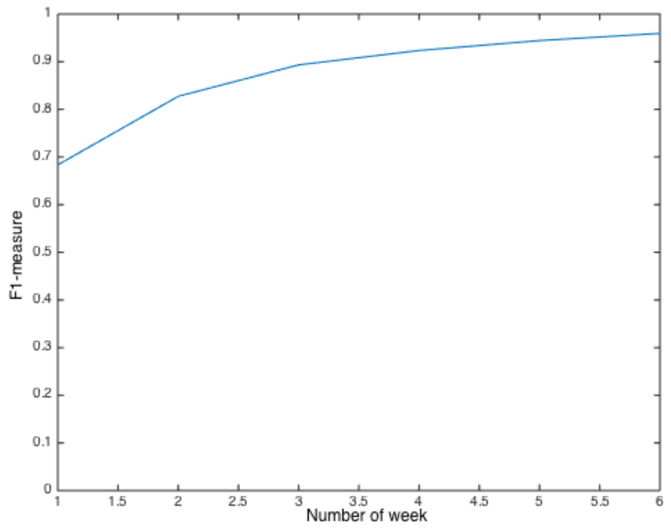


Figure 3.6 – Accuracy of the prediction against the number of weeks of observation

We found that the performance is satisfactory, even when predicting with only the events for the first week (F1-score of 0.7). Furthermore, the prediction accuracy increases sharply as we increase the number of weeks of data available. For example, 0.89 at the end of the third week and 0.94 after 5 weeks. This shows that the algorithm adapts to the course fairly quickly. The fact that the cluster can be inferred efficiently indicates that Semi-Markov Models describe well the behavioural patterns of students (**RQ1**).

3.4 Conclusion

In Section 3.3 we presented a probabilistic generative model that can be used to simulate activities of MOOC students. Our model is based on a set of four possible activities for the students:{`VideoPlay`, `Submission`, `ForumView`, `ForumPost`} to which we added an `EndOfDay` event. We used the Expectation-Maximisation algorithm that allows us to compute parameters of a probabilistic model and perform clustering of the students depending on their behaviour characteristics. The algorithm is built in a general manner and can be applied to models of varying complexity.

Firstly, we showed that Semi-Markov chains can be successfully applied to describe behavioural patterns of students **(RQ1)**. Semi-Markov chains are the direct consequence of a fully general probabilistic modeling of students' behaviour and the assumption the student behaviour at the next step can be depends only on the current step. Additionally, students are efficiently classified into clusters of behaviours described by Semi-Markov chains. Secondly, a simple analysis of three clusters that we extracted proved their potential interpretability **(RQ2)**. Finally, we used our model to infer distributions of events, provided a metric for comparing the event distributions to student data, and evaluated our simulation method **(RQ3)**. We found that our model reaches very low error rates which shows that it captures the underlying processes of students behavioural patterns.

The Homogeneity of the Markov process: The Markov assumption was introduced for reducing the number of parameters of our model. It is a strong simplification, which entails some drawbacks. This assumption implicitly requires that students behave with exactly the same transition matrix during the whole course. The motivation to keep learning should increase when getting closer to the end of the course, and, thus, the dropout rate decreases, which cannot be captured by our method. A good way to overcome this weakness is to use inhomogeneous Markov models with transitions probabilities that are functions of time.

Differences between courses: The quality of the videos, the level of difficulty of the assignments or the discussion topics in the forums are all factors that can greatly influence the behaviour of a student. None of these were included in our model. We hypothesise that adding external annotations that would impact the transition probabilities of our Markov models could help solve this problem. These do not necessarily need to be manually annotated, but could be estimated automatically. As for now, our model can be used to compare courses. For example, if we run the algorithm on two MOOCs and realise that the Video Watchers of one course have a lower engagement, that shows a lower quality of video content while differences for the Forum Follower may reveal differences on the quality of the Forum discussions.

Student simulations for experiments: Large-scale experiments are often expensive and time consuming. Although Massive Online Open Courses (MOOCs) provide a solid and consistent framework for learning analytics, MOOC practitioners are still reluctant to risk resources in experiments. In this study, we suggested a methodology for simulating MOOC students, which allows for the estimation of distributions, before implementing a large-scale experiment. To this end, we employ generative models to draw independent samples of artificial students. We use Semi-Markov Chains for modeling student's activities and Expectation-Maximisation algorithm for fitting the model. From the fitted model, we generate simulated students whose processes of weekly activities are similar to those of the real students. To our knowledge, this study was the first to propose a method to model a MOOC as a set of Semi-Markov chains. Our methodology provides a model of behaviour of students which allows us to draw samples from a distribution. Simulated students can give insights on the potential requirements of an experiment and, as a consequence, they can decrease the cost of an experiment.

The concepts of generative models in ML, and in particular Bayesian Models such as BKT are a cornerstone to the work of this thesis. In the next Chapter 4 we tackle the problem of Self-Improving learning environments and test our design using student simulations based on Bayesian student models.

4 Self-Improvement

How are learning environments becoming better over time? We discussed in the previous chapter techniques for student modeling, particularly Bayesian models. These models are used in Intelligent Tutoring System (ITS) for personalising the pedagogical content or the learning activities. It is clear that adaptivity increases the students' learning outcomes compared to non-adaptive systems [54]. In the previous chapter, we did not answer two questions: How are these adaptive systems built and how can they be made more efficient at personalising learning over time? These two questions relate to the second goal of this thesis: *self-improvement*. After teaching each student, a self-improving learning environment would learn from its interaction with the student and will seek to improve how it teaches the next students.

Adaptive learning environments are usually built with a combination of knowledge from expert educators and student models trained on collected data. For example, for an ITS using Bayesian Knowledge Tracing (BKT), experts have to decompose the exercises into skills, and the model parameters are estimated from student data (using tools such as BNT-SM [39]). Yet, expert knowledge and student data do not necessarily need to be combined to achieve personalised teaching. A human tutor adapts quite well to students even without explicitly collecting a dataset of learner data. In contrast, several approaches also try to be purely based on student data and models with no expert inputs. For example, automatically extracting skill decomposition [135] or using Deep Learning [180, 150]. Similarly, Mayo and Mitrovic compare "expert-centric" and "data-centric" approaches and propose a method that relies solely on observable variables without the assumption of hidden student states [153]. For improving such systems over time, one way would be to refine the computation of the model parameters as the system is collecting more data. In this chapter we provide algorithms which select actions for Bayesian Models and over time refine the estimation of the parameters of the models thus maximising learning outcomes for students.

Self-improvement occurs naturally in a wide variety of learning environments. For example, one can think of a professor seeking to improve their course each successive year, becoming better at teaching it, and accumulating higher quality learning material for the students. A second example could be a university which makes use of student feedback to enhance courses or the curricula. A more general practice that improves education is research and experimental studies. Such studies usually take the form of testing different pedagogical approaches in order to know which works best in which context. Instructors who run pedagogical experiments are able to reflect on their own pedagogy and enhance their teaching material for future students [252]. The research conclusions can be used by instructors or technologists to improve learning environments. These occurrences of improvement of learning environments are valuable examples to guide the design of self-improving learning environments. A learning environment is considered self-improving if it automatically improves itself, without external interventions. It is noteworthy that these three examples have as a common factor, the observation of students, which serves as the basis for improvement of the learning environment.

The field of machine learning has brought new opportunities for building self-improving systems relying on automatic data collection and data analysis. Reinforcement Learning (RL) specifically studies agents evolving in an unknown environment and seeking to improve a given reward over time. The RL-agent can use a limited number of actions sequentially, which influences its rewards and its observations of the environment. In this context, the agent must seek to better understand the environment and to maximise its rewards. This framework can also be useful to think about learning environments. The learning environment is the RL-agent, which can select among a range of activities or teaching material (RL-actions), and students are the unknown RL-environment. The learning environment must learn how students learn (understand the environment) in order to teach more optimally (optimise its reward).

The interesting particularity of RL is that the agent must balance between learning about the environment and maximising its immediate reward. This challenge is commonly called exploration-exploitation trade-off. We illustrate it with the Multi-Armed Bandit (MAB) problem [88] in Section 4.1. Exploration consists in making decisions that allow one to learn more about the environment, and exploitation consists in using knowledge of the environment to maximise the expected rewards. In the context of education, the goal of a self-improving learning environment is to select good pedagogical choices while measuring the efficiency of these choices and while testing enough different possibilities. Indeed exploring unknown possibilities is necessary to not miss on the opportunity to discover better teaching strategies. Indeed, always selecting the choice that seems optimal according to the data collected so far removes the opportunity to discover better choices.

This chapter focuses on the design of self-improving learning environments using algorithms for MAB optimisation. First, we define the MAB framework and its heuristic algorithms in Section 4.1. We then define how it has been applied to the field of education in Section 4.2. Finally, we give theoretical results on the use of MAB algorithms for Bayesian Models in Section 4.3.

4.1 The Multi-Armed Bandit Problem

The Multi-Armed Bandit is a classic problem of probability theory often used to illustrate the trade-off of exploration and exploitation. It is a special case of the more general framework of RL. The name "multi-armed bandit" (MAB) is derived from casino's slot machines called "one-armed bandit". Imagine the following situation: Alice owns ten thousand tokens that she can decide to play in sequence in any of two slot machines A and B. She also knows that the two machines have possibly different expected rewards. So far she has played three token in machine A and earned 10$ and one token in machine B and earned nothing. Should Alice decide to play all the remaining 9996 tokens in machine A from now on?

If Alice seeks to maximise her expected gains this would not be an optimal strategy because she cannot be sufficiently sure that the machine A is better than the machine B yet. In her few first trials she might have been lucky when using A or unlucky when using B. On the other hand, playing half of the tokens in A and half of the tokens in B would also, in general, not be an optimal strategy because it does not make any use of the information obtained from playing both machines.

The MAB bandit problem gives the choice between several arms numbered 1 to k that have probabilistic rewards with unknown means $\mu_1, \ldots, \mu_k$. The quality of an explore-exploit strategy S for the multi-armed bandit problem is usually evaluated using the regret as defined by Equation 4.1. The regret measures the difference between the expected reward of a strategy and the gains of the optimal arm. In Equation 4.1, $R(N)$ is the regret after N steps, $\mu^* = \max_i \mu_i$ is the average reward optimal arm and $\mathbb{E}[r_{S,t}]$ is the expected reward at step t when using strategy S. In particular, we are interested in strategies with negligible regret, which means that $R(N)/N$ decreases to 0 with a large number of steps N. Such selection strategies would, after a large number of steps, be approximately as good as always choosing the optimal arm.

$$R(N) = N\mu^* - \sum_{t=1}^{N} \mathbb{E}[r_{S,t}] \tag{4.1}$$

4.1.1 Heuristics for the MAB Problem

Although very simple to describe, the MAB problem is very hard to solve and motivates a lot of research. Gittins proposed a solution for maximising the discounted expected future reward using dynamic allocation indexes [88]. However, the solution is computationally expensive and other heuristics perform well in practice [40]. In this section, we will only detail the ϵ-greedy algorithm, the Upper Confidence Bounds algorithm and Thompson Sampling as they seem to have been the three methods most applied in the context of education.

ϵ-greedy strategy

The ϵ-greedy approach is the simplest proposed solution for MAB. The strategy consists in exploiting the arm with the highest empirical expected reward most of the time (with probability $1-\epsilon$) and exploring uniformly at random with probability ϵ. This strategy does not achieve negligible regret, and the regret after a large number of steps will always be proportional to $N \times \epsilon$. This motivates the use of lower values of ϵ, yet such low value would also mean less exploration and thus interesting arms might not be sufficiently selected until many steps. A small variation of this strategy would consist in a first phase of full exploration followed by a second phase of full exploitation. This practice would also be called ϵ-first or A/B testing [126].

Upper Confidence Bounds

The Upper Confidence Bounds (UCB) algorithm proposes an elegant solution to the MAB problem [10]. The idea of the algorithm is to consider optimistic evaluations of the different arms by adding a term that takes the measurement uncertainty into account. Equation 4.2 gives the score evaluated by the UCB algorithm. The scores are updated at each step and the arm with the highest score is then selected. In Equation 4.2, the variable t is the number of steps made by the algorithm, $T_{t,k}$ is the number of times the arm k has been selected until step t, and $\hat{\mu}_{t,k}$ is the empirically evaluated mean reward of arm k at step t. This scoring mechanism ensures that an arm that has not been tried a lot compared to other arms might be selected because it has a high value on the upper confidence bound. This algorithm achieves negligible regret over a large number of steps.

$$s(t,k) = \hat{\mu}_{t,k} + \sqrt{\frac{log(t)}{T_{t,k}}} \tag{4.2}$$

Thompson Sampling

Another heuristic for MAB optimisation is the Thompson Sampling (TS) algorithm. It has been widely recommended because it is easy to implement and performs very well in practice [40, 3]. Unlike UCB, TS is not deterministic. The algorithm consists of selecting an arm according to the probability that this arm is optimal [233, 207]. This is done by sampling from the posterior distribution of the reward for each arm and selecting the arm with the largest sampled reward. In general the rewards are modeled using Beta distributions (see example on Figure 4.1) or Dirichlet distributions (see Section 4.3). Similar to UCB, Thompson Sampling is a strategy which achieves negligible regret over a large number of steps.

4.1.2 Practical Use of MAB Optimisation

Although the problem statement appears very simplistic, solutions to the multi-armed bandit problem are not limited to a small range of use. In their book *Algorithm To Live By* [44], Brian Christian and Tom Griffiths explain how the exploration-exploitation trade-off should be used to choose which restaurant to go to for dinner, whether to spend more time with your best friend or when trying to meet new people. Within academic research, a lot of diverse fields have published on the merits of MAB algorithms. For example in the context of online advertisement [1], recommendations in e-commerce [31], user interface designs [144], or for clinical trials [233, 245].

Using high quality optimisation in educational technology can be an ethical concern. The case of clinical trials is particularly important. In this case, using a better optimisation strategy will, in expectation, select better treatments and save more lives. The same case can be made for Education. It seems desirable to make experiments to explore several teaching strategies, but not at the expense of students' learning.

4.2 Self-Improvement in Education

In this section and the following, we focus on the use of MAB optimisation in the context of learning environments. Previous research and our work in the following section motivate the use of MAB algorithms as one of the best available options for designing self-improving learning environments.

4.2.1 MAB for Adaptive Teaching

One of the uses of MAB optimisation in the context of education is for implementing adaptivity in tutoring systems. As we discussed in the thesis introduction, adaptivity requires collecting data from students in order to personalise the pedagogical content. MAB optimisation can be used to balance between giving an activity because it will reveal important information about the student or giving an activity because it is supposed to best help the student learn. For example, MAB optimisation has been used for sequencing pedagogical content based on difficulty and uncertainty about the students level of ability [209] and notably to handle the concept of the Zone of Proximal Development [47, 48]. Interestingly, another version of the MAB problem called contextual bandits allows to improve the personalisation of pedagogical activity selection [149].

4.2.2 Improving Trade-offs in Pedagogical Experiments

Outside of personalisation, using MAB optimisation for educational systems has been proposed many times [139, 138, 191, 189, 252]. The main interest is that it allows one to carry out experiments in learning environments. Running experiments is incontestably a key part of

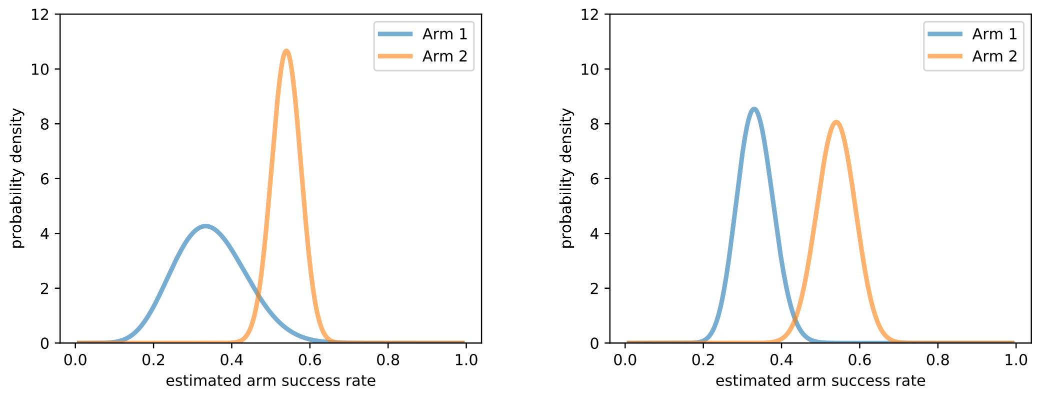


Figure 4.1 – Comparison of posterior estimates between Thompson Sampling optimisation (left) and Uniformly Random arm selection (right) with two arms of true success rates of respectively 0.4 and 0.5 after 200 throws

improving pedagogical systems. Yet, when experimenting through randomised control trials, a proportion of the study participants will always be assigned to the sub-optimal condition. Using MAB optimisation benefits participants by assigning more students to the better conditions, but unfortunately reduces statistical power of experiments [191]. Work by Liu provides a particular algorithm that can be parameterised to balance scientific knowledge and teaching quality in an educational game [139]. Additionally, it was shown that assigning optimistic priors to the MAB choices leads to increased statistical power [191].

Figure 4.1 shows the difference between using TS optimisation or splitting participants equally between two experimental conditions. The figure shows the posterior expectations of reward after 200 throws and comparing two conditions with true expected reward of 0.4 and 0.5. We can see that the Thompson Sampling optimisation has selected the best arm more often because it has lower uncertainty on the arm success rate. This is in itself a good thing if the success rate of students during the optimisation procedure matters. However, this unbalance leads to still having a high uncertainty on the estimation of the success rate of the other arm. This often would mean insufficient power for drawing statistical conclusions on this data. On the other hand, the uniform split between the two arms leads to less overall successes but a better statistical estimation of the difference between the two conditions.

In some large scale learning environments, like MOOC platforms, it seems impossible to run multiple experiments for every MOOC created because it would require too much man power to set up and analyse the results of such experiments. Beyond improving trade-offs in experiments, MAB optimisation can be used to efficiently automatise the experimental process. In general, when a learning environment contains a large quantity of pedagogical content, it becomes necessary to automatically carry out the desired experimentation for self-improvement. Using MAB optimisation would lead to several advantages, such as requiring less time from experts and improving students' learning outcomes.

4.2.3 The Cold-start Problem

MAB optimisation is relevant to support decisions in a learning environment with insufficient information. The cold-start problem in machine learning occurs when a system must make a decision without enough contextual information. It is often discussed in the context of recommender systems when a new user starts using a platform and must be recommended items [134, 80]. Learning environments face the same sort of challenge. If a new student starts interacting with a learning environment, it is likely that no information about this student has been collected so far, yet the learning environment must select pedagogical content to be delivered.

Solutions to the cold start problem usually consist of seeking different sources of data in order to build a prior about unknown students. An expert instructor who knows the pedagogical activities or teaching materials well could recommend such a prior. This expert recommendation would take the form of an estimation of the different learning outcomes of the different activities and an estimation of the expert's own certainty about the first estimation. If the expert is fully confident in his or her estimation and is reliable, then MAB optimisation would not be necessary. In the other cases, the MAB optimisation could be started from the expert's prior. Starting the MAB optimisation on different priors has been previously studied and shows that more optimistic priors lead to increased exploration in the MAB process [191]. A second possibility for designing such a prior is to use high quality simulations of student behaviour and learning processes. This can be done using student simulations, for example based on cognitive models. The simulated student would generate data allowing the system to learn a good prior on unknown students.

Even without exterior sources of data, MAB optimisation has been shown to help implement solutions to the problem of cold-start [80]. Indeed, MAB optimisation algorithms will efficiently learn from the unknown student while maximising their learning outcomes.

4.3 Applying Multi-Armed Bandits to Bayesian Models

In the several examples discussed in the previous section, MAB optimisation has been used to maximise a directly observable metric. Yet, the goal of a learning environment is the learning outcomes of students, which cannot be measured directly but have to be inferred from observations. This task, as we detailed in Chapter 3, can be done by Bayesian student models. For example, BKT infers the mastery of students from observations of their performance [55]. In the case of BKT, maximising the performance is equivalent to maximising the outcome on students' knowledge states. However, this equivalence cannot be generalised for all learning tasks and all student models. In particular, we will analyse the process of inductive reasoning in Chapters 5, 6, and 7 for which observations of students are used to infer their states but are not desirable to be maximised directly. Moreover, we illustrate this aspect with a simple example below.

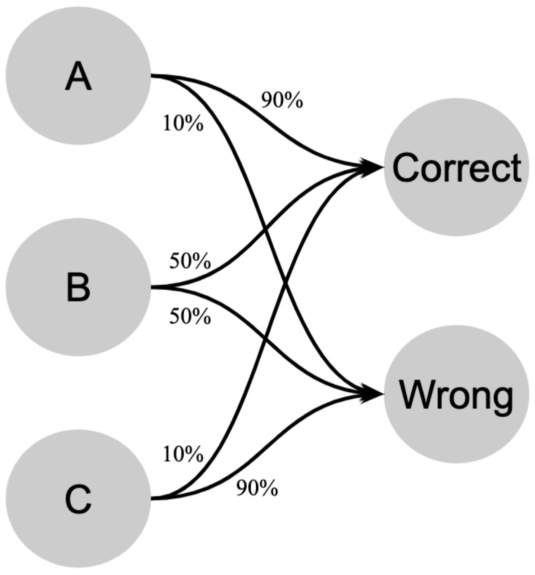


Figure 4.2 – Example of student states (A, B, and C) and probabilities of observations (correct and incorrect)

This motivates the work presented in this section. First, we show a simple example where maximisation of a given observation fails to guide students to desired knowledge states. Then, we show in the case of Bayesian Models how to sequentially estimate the transition probabilities from observations. Finally, we show how to use these estimations to select different learning activities while handling the exploration/exploitation trade-off using Thompson Sampling. We finally confirm our theoretical result with a numerical simulation.

Illustrative example: A teacher wants the students to learn how to multiply two 1-digit numbers and gives, as a test, the computation of 3×7? with the choices 20, 21, 22, 23, 24, 25, 26. Now consider two activities: 1 and 2. Activity 1 has a small chance (30%) to correctly teach students how to do multiplication. Activity 2 teaches students that multiples of 3 must have digits that sum to 3, 6 or 9. By applying the rules learnt in activity 2, students would be able to eliminate most of the choices and answer correctly half of the time. Maximising the probability of a correct answer leads to preferring activity 2 to activity 1. Yet, the teacher does not much value the state in which students are after activity 2 and would care more about the 30% chance of correctly teaching multiplication procedures. This example is depicted in the graphical model in Figure 4.2. If the teacher does not value the state B, but there exists an activity that is very efficient at making students transition from C to B, then maximising the expectation of a correct answer can be counter productive.

4.3.1 Problem Definition

We use the same definition and notation for Bayesian Models as presented in Chapter 3.

- A set of the possible states of the student: $s \in \mathscr{S}$.
- A set of possible observations: $o \in \mathscr{O}$.
- A set of learning activities: $a \in \mathscr{A}$.
- Probabilities of observations depending on the students' states: $P(o|s)$
- Probabilities of transitions between students' states depending on the chosen activity $P(s'|s,a)$, the probability for the student to be in state s' after being in state s and doing learning activity a.
- A function of preferences of the teacher about the states of the students characterised by a utility function $u(s)$.

The optimisation process should find the activity a^* that maximises the expected utility $U(a)$ as described in Equation 4.3.

$$a^* = \max_a \sum_{s \in \mathscr{S}} u(s) * P(s|a) \tag{4.3}$$

4.3.2 Estimating Transition Probabilities from Observations

In this section, we focus on the problem of estimating the transition probabilities of several unknown learning activities when the states and observation probabilities of the Bayesian Model are known. Because the transition probabilities from a given state is a multinomial distribution, the uncertainty about the parameters can be described using Dirichlet distribution, which is the conjugate prior of the multinomial distribution [192]. The Dirichlet distributions are a natural choice for modeling uncertainty about the model parameters, because it is the conjugate prior of categorical probability distributions that are used by Bayesian student model for the transition probabilities [19, 17].

The estimation relies on modeling the probabilities of observations. The observation probabilities of a given activity $P(o|a)$ can be obtained from the transition probabilities and the observation probabilities from states (see Equation 4.4). This equation can be written as a matrix product. O_a is the vector of values of $P(o|a)$ for all observations, O_s is the matrix of value $P(o|s)$ for all observations and states, and T_a is the vector of transition probabilities.

$$O_a = O_s T_a \tag{4.4}$$

Equation 4.4 gives us the first necessary condition for being able to estimate probabilities of transitions, which is that the matrix O_s is full rank. In this case, O_a is uniquely determined by the transition probabilities. On the other hand, if O_s is not full rank, different sets of transition probabilities could lead to the same probabilities of observations and, thus, would not be distinguishable. This problem has been called *identifiability* in previous work [19].

A simple example of non-identifiability is given with the graphical model of Figure 4.2. In that example, it is not possible from the observations to differentiate the difference between a probability 1 of transition to state B and a probability 1/2 of transition to both A and C. In this case, the system should be improved by enhancing the mechanism for observing students. Analysing more precisely what type of incorrect answers are given could help solve the problem on identifiability (see illustrative example at the beginning of the section).

If we assume a uniform prior on all possible transition probabilities, after observing the outputs $n_1, \dots, n_k$ times each of the observations $o_1, \dots, o_k \in \mathcal{O}$, the posterior on the probabilities $P(o|a)$ is equal to a renormalised Dirichlet distribution as in Equation 4.5. A renormalised Dirichlet distribution is necessary because only the distribution of observations that result from transition probabilities in the Bayesian Model are possible. For example, in the simple case of BKT, it is not possible to have a probability of exactly 100% of correct answers, because the probability of a correct answer will always be clipped between P_G and $1 - P_S$ ($P_G \leq P(correct) \leq 1 - P_S$) by the structure of the Bayesian model. The posterior on observation probabilities then allows us to compute the posterior on the transition probabilities by inverting Equation 4.4 in the condition of identifiability.

$$P(o|a, n_1, \dots, n_k) = Dirichlet^{(renormalised)}(n_1, \dots, n_k) \tag{4.5}$$

4.3.3 Optimising Activity Selection with Thompson Sampling

According to the optimisation criteria of TS, a learning activity should be chosen with probability equal to the probability that this activity is optimal (see Equation 4.6). This can be done by sampling from the posterior distribution of observations and inferring the corresponding transition probabilities and average utility of the chosen activity. This requires one to sample from a renormalised Dirichlet distribution, which has been shown to be computationally expensive [85]. Instead, we propose an approximation of our model using a usual Dirichlet distribution. After a sufficient number of observations, the Dirichlet and renormalised Dirichlet distributions will be arbitrarily similar. Thus, we expect this approximation to not impact the optimisation performance. We compare both methods with numerical simulations in the next subsection.

$$P(a_i) = P(\mathbb{E}[U(a_i)] = U(a^*)) \tag{4.6}$$

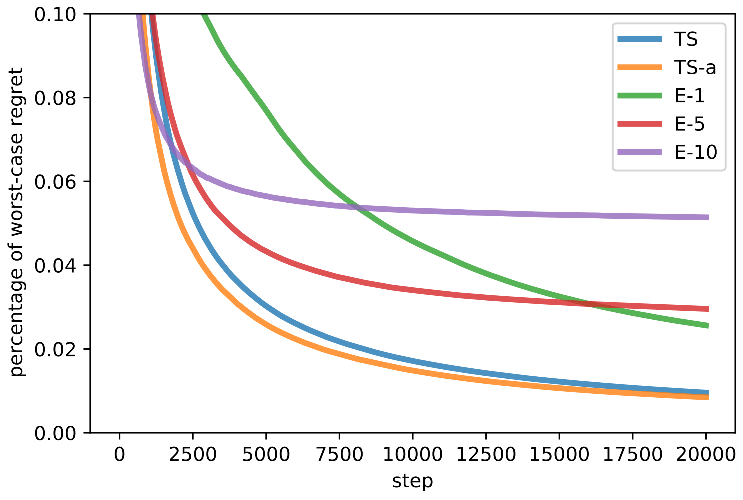


Figure 4.3 – Average regret of 5 strategies for optimising the selection of learning activities with a Bayesian student model. The regret is averaged from 200 repetitions of the simulation of 20000 students.

4.3.4 Numerical Simulation

Our optimiser uses TS based on Equation 4.6. The optimisation procedure is only aware of the observation probabilities from the different states and the utility value of each state. The optimisation process then only has access to the sequence of observations resulting from its choices. The transition probabilities must be estimated in order to compare the utility of the different possible learning activities.

To test our proposed method, we generated simulations of students using the Bayesian models depicted in Figure 4.2 and several MAB optimisation strategies. Simulations were repeated 200 times and ran through 20000 steps. We compared TS using the exact posterior distribution and our proposed approximation with the ϵ-greedy optimisation strategy using three different parameters (1%, 5% and 10%). Figure 4.3 shows the evolution in time of regret. We found that exact TS and the approximation had very similar regrets and were both superior to all versions of the ϵ-greedy strategies both in terms of quickly reaching lower regrets in the first optimisation steps but also in terms of continuously decreasing the average regret with more optimisation steps. As a consequence, we have shown that our proposed optimization strategy is superior to non-adaptive exploration-exploitation methods. Thus, implementing such optimisation in a learning environment will beneficial to students' learning outcomes.

4.4 Conclusion

In this chapter we have described the probabilistic problem of Multi-Armed Bandits for which heuristics are very useful solutions to the Exploration-Exploitation trade-off. We discussed how MAB optimisation is used in education, notably for improving outcomes for students during pedagogical experiments, but also to automatise experimentation of teaching content, which is a core need of self-improving systems. MAB optimisation can also be used to alleviate the cold-start problem or to implement adaptivity in tutoring systems.

In Section 4.3, we showed a simple example where maximising the probability of an observation is detrimental to teaching. Instead, we recommended to maximise the probability of transitions to desirable states. We extended the usage of MAB to Bayesian Models by using Dirichlet distributions to model the uncertainty about the observation and transition probabilities of different learning activities. Finally, we validated our proposed implementation with an experiment with simulated students. Our proposed algorithm can be directly used in any learning environment using Bayesian models and trying to estimate the transition probabilities of unknown pedagogical content. In particular it is relevant to be combined with the type of student models that we described in chapters 3, 6, and 7, in order to design a self-improving learning environment.

In conclusion, this chapter laid the theoretical foundations and algorithmic tools necessary to the design of a self-improving learning environment. The main challenges of such a learning environment is to not rely purely on an adaptive student model but also to assume that the student model would be improved by collecting useful student data. Yet, pure data collection of the type done in randomised trials can be detrimental to students' learning and MAB optimisation offers a more nuanced and subtle strategy. Additionally, our contribution includes a generalisation of the MAB framework to adapt to the case of Bayesian student models, which are a widely used tool in adaptive learning environments.

5 Inductive Teaching

In the previous chapters, we have provided contributions on student models and self-improving algorithms. The computational methods we used are not specifically targeted to any specific domain of education. As explained in Chapter 1, we wish to contribute to the educational practice of inductive teaching. In this chapter we report on a study analysing several aspects of inductive reasoning. In the following chapters, we provide computational methods for adaptive and self-improving learning environments for inductive teaching.

Induction is a form of reasoning that consists of drawing conclusions from observations. A simple example of inductive inference is to conclude that "all crows are black" after seeing many black crows and no crow of another color. Induction is usually defined in opposition to deduction, which starts from premises and applies logical rules of reasoning to reach new conclusions. A deductive approach to the previous example would take the form of deducing from the premises "all crows are black" and "this bird is a crow" that "this bird is black". Deduction and induction are opposed in several ways. Induction is subjective and its conclusions are in general probable and not certain [227]. On the other hand, outside of reasoning errors and inaccuracies, deduction is objective and the truth of its conclusions is ensured given the truth of the premises.

These differences in reasoning interestingly lead to the practice of inductive teaching, which diverges from the more common deductive teaching. With a deductive approach a teacher would usually start by teaching about abstract concepts and rules before guiding students to analyse examples and solve problems. The inductive teaching approach, on the other hand, consists of firstly showing examples to students and to let them think without having been taught the theories beforehand. Inductive teaching is part of a large family of pedagogical approaches that aim to support the students in constructing or acquiring knowledge by themselves [95]. These approaches take many forms such as inquiry learning, discovery learning, problem-based teaching, or case-based teaching [102, 185]. With deductive teaching, the teacher is at the center and must transmit the knowledge directly to students while in inductive teaching, the teacher acts as a facilitator to the learning process and the knowledge creation originates from the students [223].

Inductive teaching is often superior to deductive teaching in terms of students' learning outcomes [204, 215], but it can be difficult for teachers for two main reasons. It requires a high level of preparation to choose the teaching materials, and it needs more classroom time to support students throughout the inductive process. Adaptive learning environments such as intelligent tutoring systems can potentially facilitate this work for teachers by supporting the automatic adaptation of examples based on a student's knowledge and inductive reasoning ability.

In this chapter, we will first detail aspects of inductive teaching practices in Section 5.1. Then, in Section 5.2, we focus particularly on analysing students' inductive reasoning and approaches to model it. In Section 5.3, we present a study with 222 students on a categorisation task where participants must guess a classification rule by observing examples. In this task students' prior biases can impact their learning and behaviour. The study specifically analyses how students differ on the criteria they use to classify examples into categories and on their flexibility in their choice of criteria. In Chapter 6 we will use the data collected in this study in order to design a knowledge tracing algorithm for inductive reasoning.

5.1 Inductive Teaching

Inductive teaching is motivated by the idea that learning comes from exploration and discovery [5]. Students reach better learning outcomes (for examples, in terms of conceptual understanding or memory retention) when they discover the knowledge by themselves. Quoting Seymour Papert and Idit Harel [95]: "*Instead, I must confine myself to engage you in experiences (including verbal ones) liable to encourage your own personal construction of something in some sense like it. Only in this way will there be something rich enough in your mind to be worth talking about.*" This quote boils down to the idea that students will understand and remember more in depth the knowledge that they were able to construct through experiences. Beyond letting students discover theories by themselves, inductive teaching also helps to motivate why the general theory or knowledge is needed [208] and also to prepare students to learn [204].

Inductive teaching is part of a larger family of pedagogical approaches that aim to support the students in constructing or acquiring knowledge by themselves. Such teaching practices have taken many forms [185]. **Inquiry learning**: the students are given questions to answer or observations to explain before receiving instruction; **Problem-based learning**: the students must try to solve open-ended problem (often done in groups); **Case-based teaching**: the students must analyse a situation given to them; **Just-in-time teaching**: the students complete assignments just before the class which the teacher uses to adapt the course; **Discovery learning**: the students discover the conceptual knowledge by themselves in the process of solving an assigned task. Researchers have developed scaffolding for activities such as problem-based learning and productive failure [64, 104].

Additionally, through inductive learning, students not only learn about the topic of interest but also practice methods for discovering information by themselves [102]. Practicing inductive reasoning gives students a very valuable skill that they will be able to transfer to other domains, which require to balance evidence and generalisation. It was argued for example that inductive teaching is superior to deductive teaching for preparing students to become software engineers [208, 120]. A software engineer has to face a problem, analyse it, and only then seek the knowledge required to solve it. It seems unrealistic to teach software engineering theories without motivating the instruction with examples. The importance of inductive reasoning has also been promoted in the context of teaching statistics and data science, which are exploratory in nature [96].

During inductive activities, students need to activate their prior knowledge to figure out new solutions to the task they are given [204, 205, 143], and they need to be able to make comparisons and identify the key features of the problem [114]. Additionally, students learn more when they are confronted by impasses in which they need to change and update their mental model [143]. Students do not naturally engage in these learning processes and the support that is needed can depend upon the current state of the student. For example, in a classification task, if students only ever receive feedback that supports their mental model, they will not change their point of view. However, how much students change their point of view from feedback may depend upon the student. In this case, learning analytics can be used to better understand the current state of the student so that feedback can be provided that may be most useful to help the student transition to a new state. In Section 5.3, we analyse the biases that students may bring to inductive example categorisation tasks.

5.2 Inductive Reasoning

Human ability to infer rich knowledge from a very small number of examples has fascinated researchers in the fields of education, cognitive science, statistics and machine learning [231, 130]. In machine learning, the task of inferring categories from a few examples has been called one-shot learning and is strongly guided by the analysis of human inductive reasoning mechanisms [203]. We describe in this section several of these models.

5.2.1 Reasoning About Examples and Categories

Approaches to model student inductive reasoning consider students' identification of similarities and dissimilarities between concepts [45]. Tversky's Contrast Model of Similarity [235] relies on sets of features attributed to two objects being compared and explains several effects in inductive reasoning [98]. Equation 5.1 describes how the similarity between two objects a and b is computed, according to Tversky's model, based on their sets of features A and B. The similarity contains a positive term for $A \cap B$ which counts the common features of a and b and two negative terms for $A - B$ and $B - A$, which are respectively the features that a has and b does not and the features that b has and a does not. Interestingly, the similarity between a and b, according to Tversky, is not necessarily symmetrical. For example, one would make the distinction between reasoning about how features of birds apply to crows and how features of crows apply to birds.

$$s(a,b) = \theta f(A \cap B) - \alpha f(A - B) - \beta f(B - A) \tag{5.1}$$

Another approach to model inductive reasoning is the Similarity Coverage Model [168]. This model describes how convincing are generalisation arguments from examples to category. For example, the argument, "Elephants and rhinos are grey; therefore, all mammals are grey." is less convincing than, "Elephants and mice are grey; therefore, all mammals are grey." because "elephants and mice" covers the category of mammals better than "elephants and rhinos".

The notion of relevance of features is also a key concept to model induction. The relevance framework defines the relevance of a feature for some category [155]. In simple terms, a feature is relevant if the category has it but the majority of other categories do not have it. The relevance of a set of features can be defined similarly, if a category has all the features from the set, but different categories do not have all the features simultaneously. Different individuals exhibit differences in their reasoning about feature salience either based on culture [43] or expertise [50]. Additionally, it was shown also that when given the task to learn the names of categories of objects based on features like color or shape, children were not only able to learn the categories but also to learn which features are more important than others for naming objects which allows for faster learning in following tasks [225]. This last example shows the ability of learning to learn, which is also one of the main techniques used for one-shot learning in ML [130].

Bayesian Inference

Within modeling of human inductive reasoning mechanisms, a particular type of model stands out, which consists of modeling induction as a form of Bayesian reasoning [231]. Bayesian reasoning assigns probabilities to hypotheses according to two mechanisms [178]: the *prior probability* $P(h)$ of an hypothesis and the *likelihood* of observations $P(o|h)$. These two terms are then combined to compute a posterior probability $P(h|o)$, which defines how the level of belief in the hypotheses h_i has changed after observing o. The posterior is computed according to Bayes rule as given by Equation 5.2.

$$P(h_i|o) = \frac{P(o|h_i)P(h_i)}{\sum_j P(o|h_j)P(h_j)} \tag{5.2}$$

Several studies have revealed many similarities between human inductive reasoning and Bayesian inference. For example, Bayesian inference can explain how one can learn a concept from a small number of examples [231] or learn categories from only positive examples in the case of hierarchical categories [232]. Modeling inductive reasoning in terms of Bayesian probabilities can also explain several aspects of induction about categories and properties in a similar fashion as the Contrast Model of Similarity and the Similarity Coverage model [99].

Such models create an inductive bias (*prior*) that they use to make inference about categories, as is required for Bayesian inference [130]. For example, Lake used Bayesian Program Learning and reached learning capabilities similar to human induction on recognising new alphabet letters from a few examples [130].

5.2.2 Reasoning About Uncertainty

In deduction, the truth of the premises ensures the truth of the conclusion. This is not the case for inductive reasoning, which tries to draw conclusions when the observations are evidence for the truth but not sufficient for absolute certainty. If three fruits taken at random out of a bag of fruit happen to be apples, is it sufficient information to conclude that all the fruits from the bag are apples? A purely deductive approach would require one to make sure every single fruit in the bag is an apple before reaching that conclusion. The inductive approach would be to conclude with a reasonable amount of uncertainty that it seems that the bag contains only apples, but there is a small chance that it might not be the case.

Uncertainty in the context of inductive reasoning is another reason why models using Bayesian inference explain the process of induction well [231]. As described by the *likelihood* term in Bayesian inference, observations during the inductive process are used to update the prior belief into the posterior. Certainty is only reached when the likelihood term changes the posterior into a probability of 1 or 0. These examples tell us that it is meaningful to consider that students make probabilistic decisions in the context of inductive learning. However, researchers have shown that people often deviate from exact Bayesian inference [113]. For example, people may reason about the single most likely category and fail to account correctly for the uncertainty of several categories [97].

5.3 Experiment: Individual Differences and Flexibility in Inductive Reasoning for Categorisation of Examples

These models of inductive reasoning motivate our analyses of several more aspects of the process of categorisation of examples. Notably, our goal is to enable the design of models, similar to the models used for knowledge tracing, for the context of inductive teaching activities. Our experiment seeks to reveal which features students are able to perceive and use to categorise objects, how these features generalise from a topic to another, and, most importantly, how feedback influences students. In this section, we describe an experiment with 222 participants, and give a statistical analysis of the data collected. The data from this experiment also serves as the basis for designing a Bayesian Model of student inductive biases in Chapter 6. More

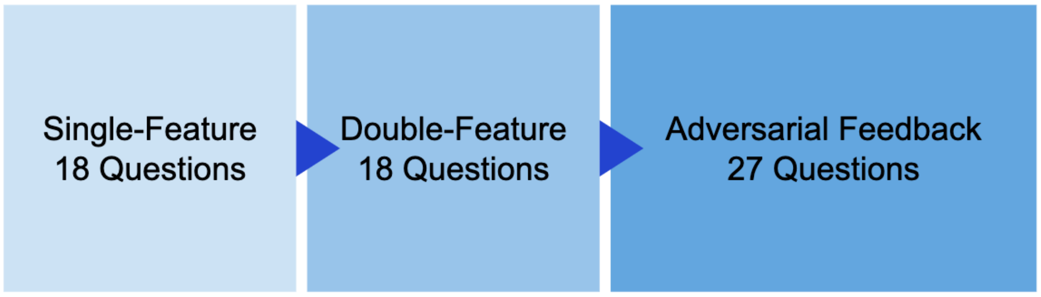


Figure 5.1 – The three subsections of the inductive activity: feature recognition (Single-Feature questions), feature preference (Double-Feature questions), and feature preference flexibility (Adversarial Feedback questions).

precisely, we analyse the following research questions:

- **RQ1**: What features impact students' decisions when students are asked to classify an object into a category? Our hypothesis is that students will have an inherent bias that drives how they choose the features with which to answer classification questions and that these biases will vary for individual students.
- **RQ2**: Are students' feature biases topic independent or topic specific (for example, categories based on the geometrical shape color and categories based on animal color)? Our hypothesis is that students' choices on one topic will be predictive of students' choices on another topic for similar features.
- **RQ3**: How do students change their biases when receiving feedback? Our hypothesis is that students will show some resistance to change and that this resistance will vary between individual students.

5.3.1 Methodology

In this section, we describe our learning activity. This activity has been designed with the purpose of testing all three research questions previously mentioned. In particular, the activity is divisible into three parts that we describe here (see Figure 5.1). The first part provides the students with a brief introduction to the task and allows them to become familiar with the concepts. The second part aims at answering RQ1 and RQ2, and the third part aims at answering RQ3. Additionally, we describe the study set-up that we used for data collection.

Learning Context

For the inductive activity, the students were asked to engage in a computer-based categorisation activity. The activity consisted of a series of 63 independent questions, which can be answered rather quickly. The questions took on average six seconds to be answered with a

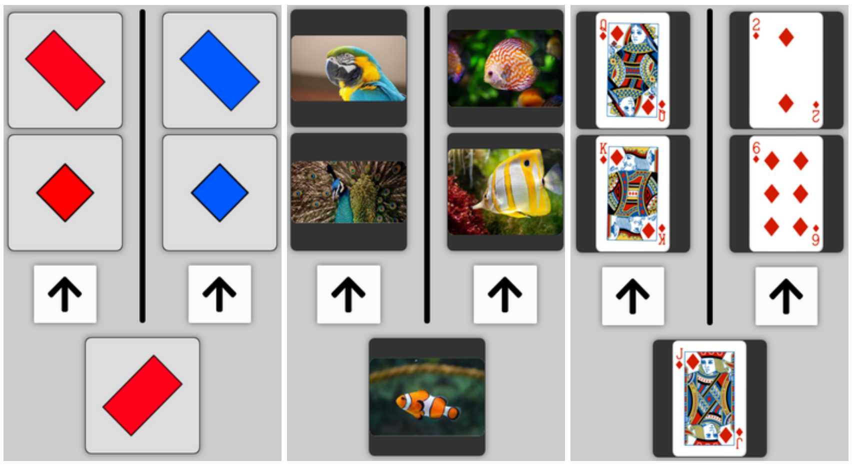


Figure 5.2 – Examples of questions from the introduction section. The topics are from left to right: Geometry, Animals, and Cards.

standard deviation of six seconds. Only a small proportion of the students needed more than 10 minutes in total to complete the activity. Each question contained four images divided into two categories and a fifth image that had to be classified into one of the two categories (see Figure 5.2). The goal of the exercise was for the students to identify common properties within each category of examples and inductively guess a rule that explains the separation and use this rule to classify the fifth image.

The 63 questions were evenly divided into three topics: Geometry (G), Cards (C) and Animals (A). For each topic, three features were identified and systematically applied to the topics. Additionally, the questions were divided into three distinct sections: feature recognition, feature preference, and feature preference flexibility. Each of these sections is described in more detail below.

For each of the three topics, the activity questions were designed around three common types of features. The three types of features that we selected covered the orientation, the type and the color of the examples. Table 5.1 describes the categories based on the general features for each of the three topics.

Structure of the Activity

The 63 questions of the activity consisted of three distinct parts, feature recognition, feature preference, and feature preference flexibility, although this differentiation was not displayed to the students. For all of the students, the order of the sections and subsections, described in more detail below, remained the same. However, within any given subsection, students received the same questions in a randomised order.

The first section, feature recognition, consisted of 18 questions, which were divided into two blocks of nine questions. Each of the blocks consisted of one question for each combi-

Geometry (G)		
Color (C)	Type (T)	Orientation (O)
Green, blue or red shapes (see Fig. 5.2)	Squares and rectangles or parallelograms with no right angle	Shapes are either horizontal and vertical or rotated by $\frac{\pi}{4}$ rad
Animals (A)		
Color (C)	Type (T)	Orientation (O)
Colorful animals and grey or black and white animals	Birds, fish, or feline (see Fig. 5.2)	Animals moving to the left or to the right
Cards (C)		
Color (C)	Type (T)	Orientation (O)
Red and black cards	Figures or numbers cards (see Fig. 5.2)	Vertical or horizontal cards

Table 5.1 – Description of the features used for the examples in the application

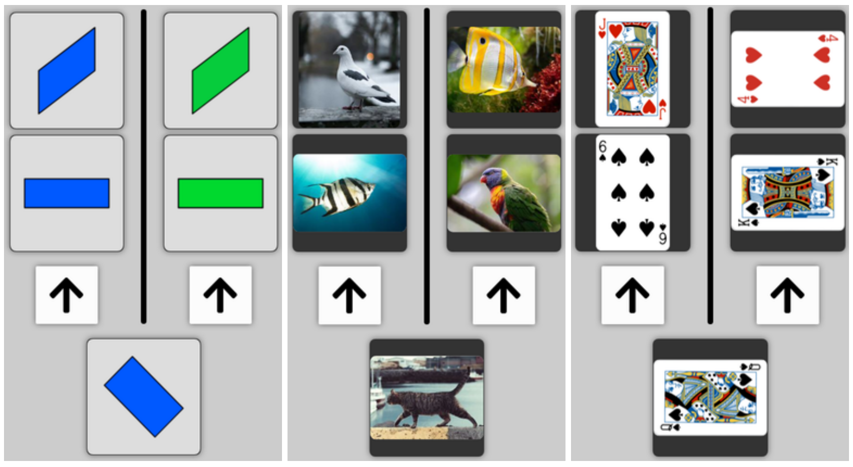


Figure 5.3 – Examples of questions with mixed features

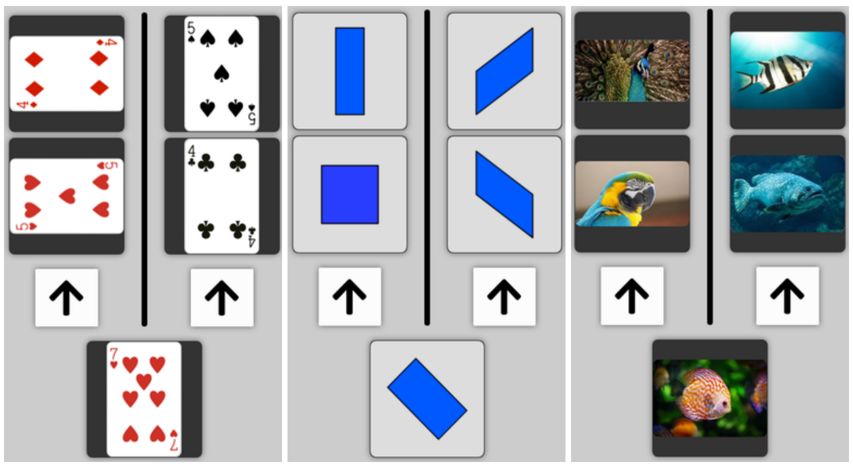


Figure 5.4 – Examples of ambiguous questions testing students' feature preference. From left to right: Color or Orientation; Orientation or Type; Color or Type;

nation of topic and feature (see Table 5.1). Each of the questions had a correct answer for the categorisation of the fifth card. This section allowed us to assess the students' feature recognition.

For both blocks, the students received correctness feedback as either a green check or a red cross, informing them if they put the item into the correct group (see Figure 5.5). In the first block, they also received a text explanation of what the correct feature was for the grouping. We gave this feedback in both instances so that if students had the correct category based on guessing or an incorrect feature, they would still have the same information as the students who had the incorrect category. The second block also provided feedback but limited to the green check and red cross without textual explanation. Furthermore, in the second block the questions were made harder by not keeping the other two features fixed even though only one feature correctly separated the two categories (see Figure 5.3). In this case we believe it is more difficult for the student to identify the correct property because varying features can distract from the feature which is correctly explaining the categories.

The second section of the activity is the feature preference, which consisted of 18 questions that were also divided into two blocks of nine questions. Unlike the first section in which the students had to recognise a given feature, in this section, the fifth example could fit into either group based upon which feature was being used to classify the item. In other words, for these 18 questions, the possible answer was not unique but instead reasoning about a specific feature was made to produce a different answer than reasoning about another feature (see example of a question in Figure 5.4). In this way, we expected students to select the group the item belonged to based on which feature they found more relevant. Again, within each block there was one question for each pair of topic and feature comparison. For this section, all student answers received a green check as positive feedback. Similar to the feature recognition section, the questions of the second block were made harder by not keeping features unrelated

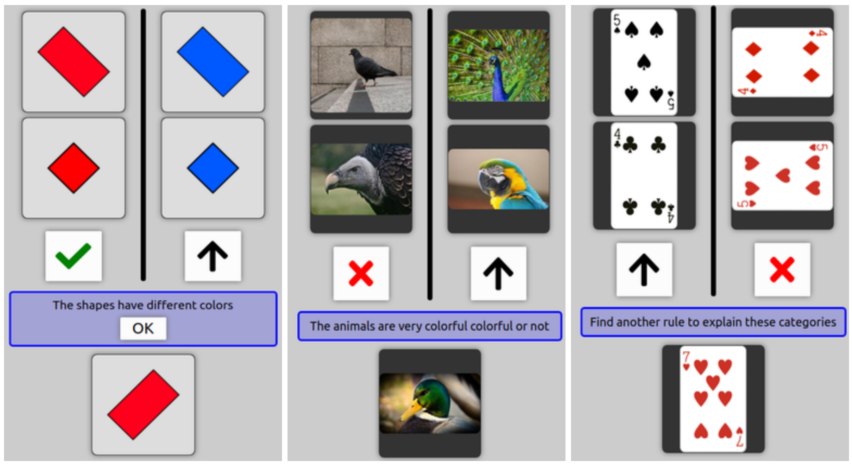

The shapes have different colors
OK
The animals are very colorful colorful or not
Find another rule to explain these categories

Figure 5.5 – Examples of feedback given to students. From left to right: correct feedback in the introduction section; incorrect feedback in the introduction section; adversarial feedback in the preference flexibility section.

to the current question constant.

The questions from this second section were designed to evaluate students' feature biases. Furthermore, the use of similar features across different topics was designed to detect whether a bias for a specific feature in one domain would correlate to the same bias for another topic.

The final section of the activity is the feature preference flexibility section consisting of 27 questions. This section follows a structure that uses three types of questions to gauge how flexible a student's preference is for a given feature. First, the students were given a comparison question similar to the questions they received in the feature preference section. However, the students were given negative feedback for their answer independent of the answer selected. We call this feedback "adversarial feedback". The adversarial feedback was shown as a red cross indicating that the student answered incorrectly even though all answers can be explained by reasoning about a particular feature. Additionally, the students received a prompt asking them to think about another explanation for answering the classification question (see Figure 5.5). This question is directly followed by a question on the same topic and feature comparison in which the student can again place the item in either group. For this second question, the feedback given is always positive. Finally, after the student receives a combination of each topic and pair of features for both question types, they receive a delayed comparison question for each topic and feature combination with positive feedback. This pattern followed the structure shown in Figure 5.6, which was repeated three times in order to cover all nine possible combinations of topic and feature comparisons.

The adversarial feedback questions were designed to make the students change their feature biases. As a student answers a question based on a specific feature and receives negative feedback, we expect the student to either answer the following question on the same feature comparison differently or possibly resist change. Furthermore, we used delayed questions

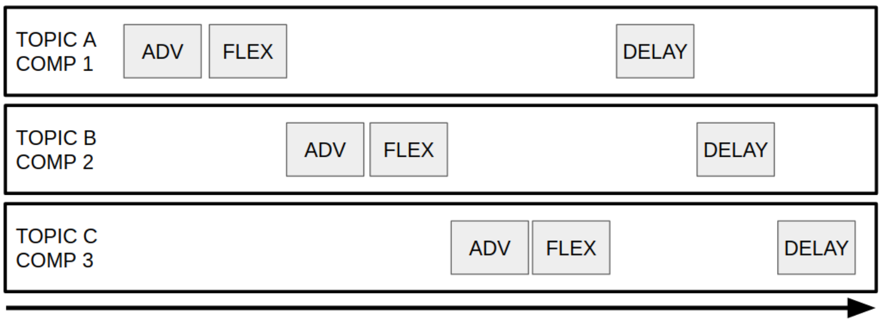


Figure 5.6 – Structure of the feature preference flexibility section. (ADV stands for questions with adversarial feedback, FLEX for questions directly following adversarial feedback, and DELAY for questions delayed after the adversarial feedback)

on the same feature comparison in order to measure whether the adversarial feedback will diminish over time.

Data Collection

Our dataset consists of data from 222 students who engaged in the application described above. The participating students were from three different schools and engaged with the activity during their regular class time. The age of the students ranged from 8-13 ($M = 10.9, SD = 1.6$), which we split into two groups: the younger students were expected to be less familiar with the concepts being taught (ages 8-10 years old, $N = 84$) and the older students were expected to have more familiarity with the topics and concepts (ages 11-13 years old, $N = 138$). Each session with the activity ranged from 10 to 15 minutes in which all but three students completed the full activity.

The activity was distributed through a web application that the students engaged in individually using their school-provided tablet or laptop. The activity started with two questions asking the students for their preferred language (English or French) and their age. At the beginning of the activity, the students were instructed to think about a rule explaining the separation of the two categories then use this rule to decide in which category to assign the fifth image. All data was anonymously collected through a write-only database using randomly generated IDs for each student, which did not require any logging in mechanism to be implemented. A total of 13968 data points were collected across the 222 students. Each data log consisted of an anonymous student ID, the description of the current question (topic and feature or feature comparison), step in the activity (necessary due to some randomisation of the order of questions), the student selected answer (correct or incorrect for the first section and the name of the selected feature for the other two sections), and the timestamp of the transaction log. The web-application source code, and collected dataset can be found on the GitHub repository[1].

[1]https://github.com/chili-epfl/induce

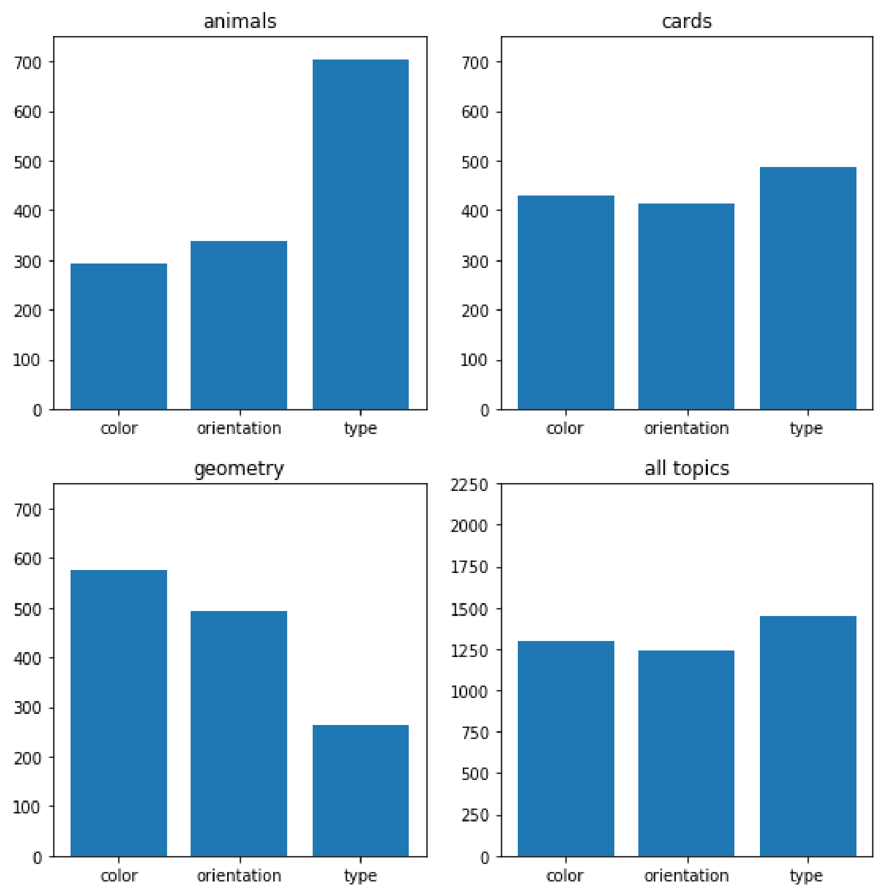


Figure 5.7 – Number of answers for each feature during the preference test section per topic.

5.3.2 Results

In this section, we will first present results related to the research questions RQ1 and RQ2 concerning which features impact students' decisions and whether these features generalise across topics.

Feature Preference Across Topics

A surprising result we observed was that students made very different categorisation choices depending on the topic. This finding reveals completely different feature biases for the three topics. Figure 5.7 shows the number of times each feature has been selected over all the questions asked to all the students. We can see that the 'type' feature was more often chosen for animal questions, less often for geometry questions and equally for the card topic compared to the other features. A Chi-Square statistical test on the categorical variables of topics and features shows a very significant difference ($\chi^2(4) = 322$, $p < 0.01$).

To analyse whether the feature biases are predictive from one topic to another, we also com-

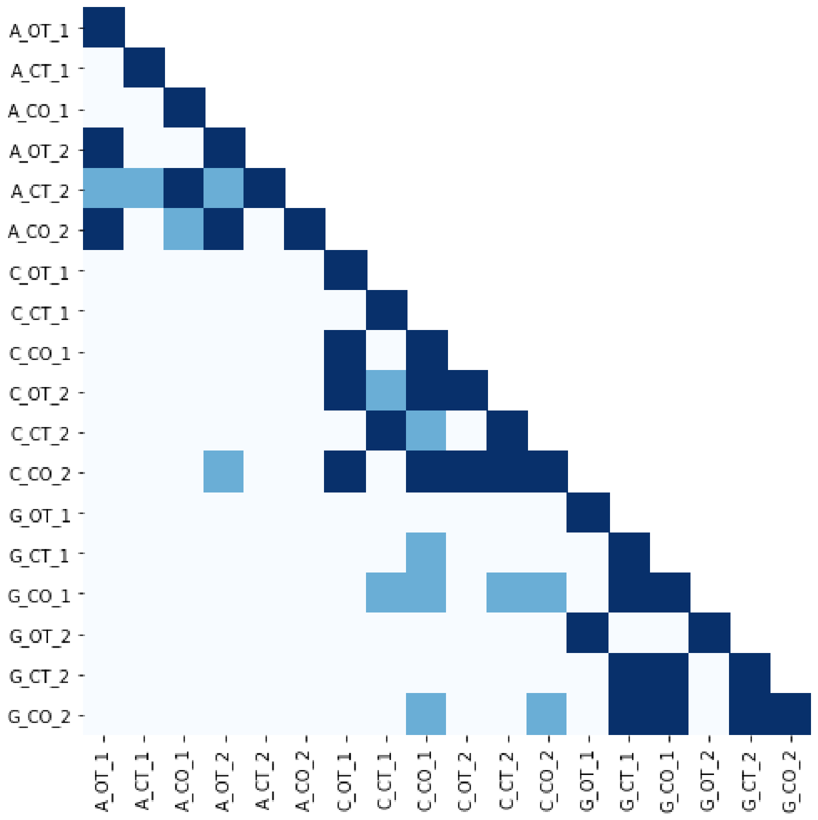


Figure 5.8 – Correlations between answers of the questions from the feature preference section (light-blue: non-significant p-value > 0.01; medium-blue: p-value < 0.01; dark-blue: p-value < 0.01 with correction). The questions are labelled as Cards (C), Geometry (G) and Animals (A) with a comparison between two of Orientation (O), Color (C) or Type (T) for block one (1) or two (2)

puted the correlations displayed on Figure 5.8. The figure shows Chi-Square correlations computed between each of the 18 questions of the feature preference section. The graph displays both significant p-values at 0.01 when adjusted for the number of tests using the conservative Bonferroni method and significant p-values at 0.01 without adjustments. We observe that for most feature comparisons, the answers given to the questions of block 1 are strongly correlated to the answers given on block 2 (7 out of 9 with adjusted p-value < 0.01). This indicates that students choose features based on their individual biases.

Only 8 of the 108 correlations tested across topics were found to be significant but only without adjusting for the number of tests. This indicates that we should not completely reject the possibility of such correlations existing. Nevertheless, we must conclude that in our context the effect size of such difference was much smaller than for the correlation within topics. An interesting note is that all but one of the across topic significant correlations are between the topics of Cards and Geometry which are more similar by their logical features than with the topic Animals. This hints at investigating further what similarity between topics would allow to transfer inductive preferences of students.

Flexibility of Students' Biases

The last point of interest of our experiment was to analyse how feedback changes students choices in answering the ambiguous categorisation questions. For this analysis we measure the percentage of students who keep their preferred feature when they answer a second question about the same comparison of features on the same topic at three moments during the activity. First, during the second part of the activity, then, right after the *adversarial feedback*, and finally, after the delay. We found a significant (p-value < 0.0001) decrease in the percentage of students who answered again with the same feature between no feedback (M = 76%, SE = 1%) and *adversarial feedback* (M = 56%, SE = 1%). This means that the feedback indeed influenced students to change their biases according to the feedback. Secondly we found a significant (p-value < 0.001) increase in the percentage of students who answered again with the same feature between no delay (M = 56%, SE = 1%) and after the delay (M = 60%, SE = 1%). This means that the effect of the feedback indeed decreased over time and students naturally went back to their initial bias, but without completely forgetting the feedback.

5.4 Conclusion

In this chapter, we first discussed the educational practice of inductive teaching which is in many contexts very beneficial to the students. We then analysed how the process of inductive reasoning is usually modeled. In Section 5.3, we described an experiment carried out with 222 students on a task of example classification. This experiment was motivated by understanding some aspects of induction, which are relevant to the design of adaptive learning environments for inductive teaching. Our results indicate that students have individual biases that drive how they choose to classify examples according to specific features and that students differ in these biases. This finding justify the importance of adaptivity in inductive teaching. We found a lack of correlations between students' biases on different topics, which means that inductive models trained on data from one topic will not generalise well to examples of another topic. Finally, we showed that students can be influenced by feedback, but still a majority are resistant to change their biases when given negative feedback.

Inductive teaching, in general, requires more effort from teachers both for preparing the lesson and for supporting students in classrooms. Building adaptive learning environments in which students are able to learn with inductive methods is valuable in this context. Such learning environments would benefit from student models allowing to trace students' biases in order to adapt the examples based on a student's knowledge and inductive reasoning ability. Chapter 6 and Chapter 7 aim to provide such algorithms.

6 Student Model of Inductive Reasoning

Decades of research have shown the impact that learner models and adaptivity can have on the learning process across a range of learning scenarios. For example, Intelligent Tutoring Systems (ITS) use student models to support students' learning processes using real-time adaptations [7] with multiple improvements occurring to these models over the years. More recently within learning analytics, models have been used for the task of process mining which seeks to describe and analyse learning processes [25]. However, the design of these models address a limited subset of learning scenarios. Specifically, many of the models developed have been targeted towards mastery learning and skill practice but less towards purely inductive approaches, such as the one we described in Chapter 5. Yet, there is a growing interest in inductive scenarios [143] as teaching with inductive methods has been shown to outperform traditional teaching across contexts [94]. Although it is still uncommon, recently, there have been a few examples of adaptive learning environments that support students in induction processes rather than supporting deduction or practice. Two examples of such systems have been used for teaching logical fallacies [65] and geography [159].

Models of inductive reasoning rarely consider differences between students that are necessary to provide adaptivity. Instead, most models aim at establishing a general description of the induction process without including individual differences. Including individual parameters has been shown to enhance student models for example in the case of BKT [258]. We should expect that individual parameters would also enhance models of inductive reasoning. Furthermore, models of inductive reasoning rarely analyse changes of the inductive reasoning approaches when students receive feedback nudging them to modify their strategy. When the goal of a model is to support a more effective learning process, how students change over time is of primary interest.

In this chapter, we intend to fill this gap. We propose a Bayesian student model of inductive reasoning that predicts students' choices for the example categorisation task described in Chapter 5. We model three aspects of induction: students' individual biases, generalisation of biases across topics and changes of inductive biases. We evaluate our proposed model on the data collected from the experiment described in Chapter 5. Our model can be used in several

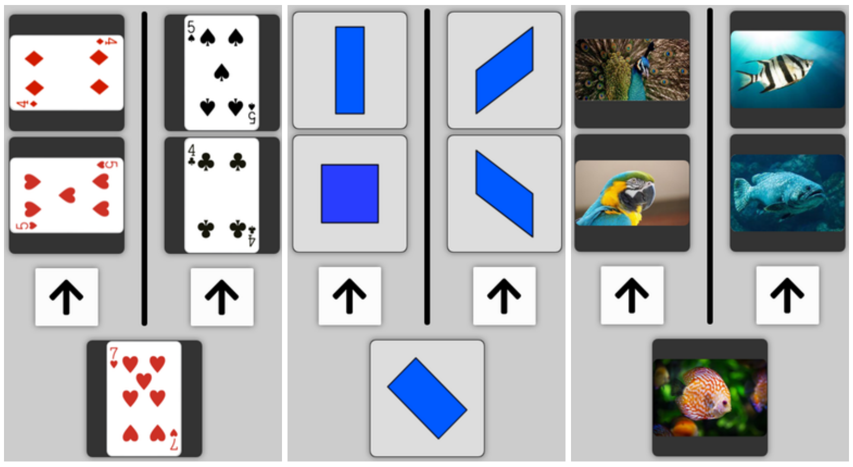


Figure 6.1 – Examples of ambiguous questions from

ways. First, the model fit can provide insights into the inductive reasoning process. Second, the model can be used to simulate students in order to help teachers design sequences of examples for an inductive learning task. Finally, our model can also be used to track students' reasoning approaches and personalise the learning environment in the context of an inductive learning activity.

In Section 6.1 we define the structure of our model and its parameters. In particular, we describe how the model is used to predict students' choices in the categorisation task (Section 6.1.3) and we describe the algorithm we use to estimate the parameters of the model (Section 6.1.2). In Section 6.2 we analyse the quality of the predictions of this model on the data from the experiment described in Chapter 5.

6.1 Definition of the Model

Within inductive learning, it is not just a matter of whether a student knows a fact or not. Indeed, in the context of induction, when making decisions there may not always be a correct answer. An example of such ambiguous questions are the questions from our experiment in Chapter 5 where several features can be used to classify the objects into categories (see Figure 6.1). In this case, to correctly model student behavior in order to provide learning support, it is important to take into account a student's bias, which may have been formed by their prior experiences. Our approach uses a Bayesian model with the hidden states representing the student biases.

6.1.1 Bias Profile Model

We used Bayesian models as we aimed to design a model that will rely on hidden states of the students about their inductive bias while only being able to observe their answers to questions.

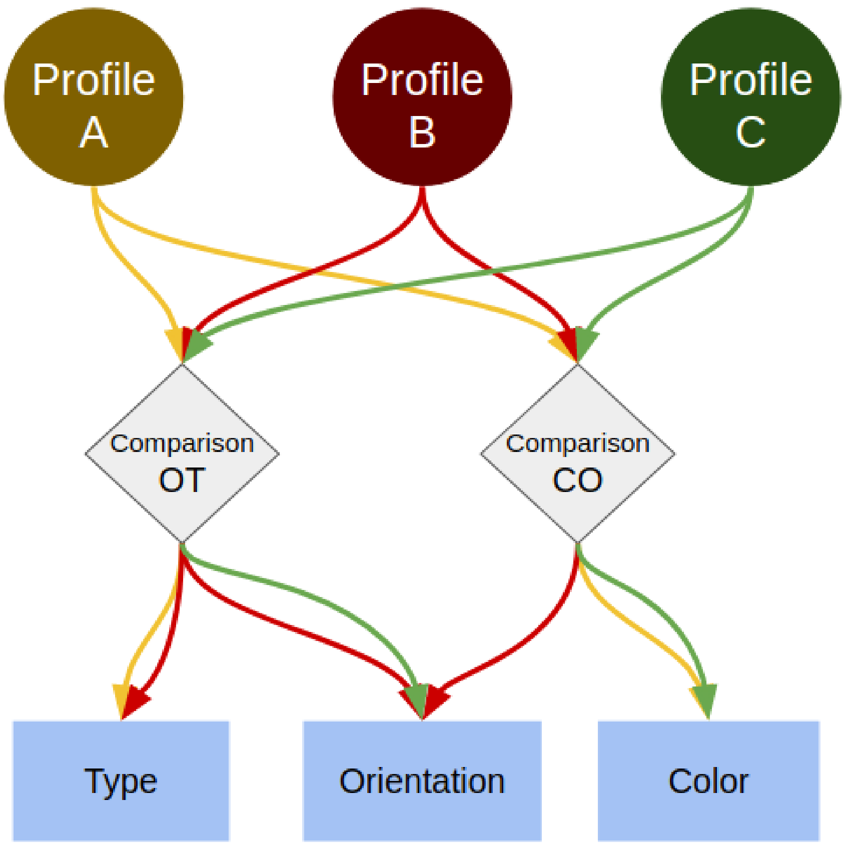


Figure 6.2 – Bayesian model of students' bias profiles.

Similar to BKT, the model for a student does not exclude possibilities of answers but instead assigns to each student the probability that the student answers A rather than B to a given question. We model a bias profile as a set of probabilities describing a student's bias towards perceiving different features. In our case, each profile contains up to nine probabilities.

Figure 6.2 shows how the bias profiles explain observations of different types of answers from the students. Given an ambiguous categorisation question on two features, a specific profile gives a probability of answering with the first or the second feature. The graph on Figure 6.2 represents our model with three types of nodes. The round nodes represent the hidden states of the students. The diamond nodes represent the actions that can be taken by the learning environment, such as selecting a specific question or giving a particular kind of feedback. Finally, the rectangular nodes are the observations that the system can collect. In the case of our activity, these observations are the features that the students use to answer the ambiguous categorisation questions.

Our model contains only one arbitrary parameter K, which is the number of distinct bias profiles to be extracted from student data. This parameter should be set to a meaningful value. The model contains several bias profiles $(B_k)_{k\in 1:K}$. Each B_k is associated with the nine parameters $p_{k,t,c}$ where t is one of the three topics and c one of the three possible feature comparisons. The parameter $p_{k,t,c}$ is the probability for a student who exhibits the bias B_k to select the first feature when given a question from the topic t and comparison c.

Bias Profile Changes

Another important aspect of our model is the possibility that students modify their bias profile while interacting with the questions and, in particular, while receiving feedback. This possibility reflects that students make progress toward learning the features that align with the learning goals. We model the possibility of bias change for the questions following each adversarial feedback and another bias change for the delayed feedback. This is done by computing a new probability distribution that more accurately explains the student's answer for each bias profile after the feedback. The amount a student changes can indicate their level of bias flexibility. Additionally, we also compute a second Bias profile change to account for the delay after the feedback. For the flexibility and delayed questions, two other sets of parameters are computed for each bias profile B_k, respectively $p'_{k,t,c}$ and $p''_{k,t,c}$, which are used in the same way as $p_{k,t,c}$.

Feature Perception

We do not include in our model the ability for students to perceive a given feature in simple categorisation questions. The first section of the activity was developed to teach the students about each of the possible features. For this section the answers of students are correct or incorrect and we can easily decompose the questions into a set of skills. In this case, BKT [55] is a good modeling approach as the student either knows the feature or not, which can be predicted based on their behavior in the learning activity. As the use of BKT is a simple application of an existing model, it is not the focus of our model. Instead, we focus on the second section of the activity where the questions are ambiguous and can be answered differently depending on students' biases. Nevertheless, our model does include the possibility that a student would never answer the categorisation questions using a particular feature. This would be characterised by the probability parameter of the student's Bias profile for this feature to be equal to 0.

Choice of the Parameter K

For the choice of the parameter K, the trade-off is that lower parameters K will capture less complexity in the behaviours of the students while a higher K will generalise less well to new students and be more sensitive to noise in the data. This is a common machine learning trade-off of model complexity versus overfitting. Additionally, given our dataset containing data from 222 students, trying to extract more than 50 bias profiles would render the estimation of the parameters very inaccurate because only a small number of students would be assigned to each profile. The choice of the optimal parameter can be done with methods such as the elbow method that we used in Chapter 3. In this chapter, we present our results using the parameter $K = 10$. The value $K = 10$ was chosen because higher values did not lead to any improvements in the predictive accuracy of the model.

6.1.2 Estimation of the Model Parameter

A very useful algorithm to estimate the parameters of the model such as our is Expectation-Maximisation (EM) [163]. The EM algorithm is specifically designed to compute high likelihood parameters with unknown latent states. For example, it has been used in the context of BKT, which also relies on partially observable hidden states of students [172]. The EM algorithm functions by iterating through two phases multiple times. First, given a set of parameters of the model, it computes the most likely distribution of the students into each of the K hidden states. Secondly, given the distribution of students, it computes the state parameters that maximise the likelihood of the sequences of observations.

The first phase consists of assigning to a student s the most likely Bias profile for that student. Equation 6.1 shows how to compute the likelihood, based on the observations of the student answers $o_{s,m} \in O_s$ ($o_{s,m}$ is 0 or 1 depending on whether the first or second feature has been selected). The second phase consists of re-estimating the values of the parameters $p_{i,t,c}$ based on the assignment of students obtained in the first phase. This is done using Equation 6.2, where $n^{(1)}_{i,t,c}$ and $n^{(2)}_{i,t,c}$ are the number of times students, who have been assigned to Bias Profile i, have answered with the first and second feature for questions on topic t and comparison c.

$$P(s \in B_i | O_s) = \prod_m P(o_{s,m} | B_i) = \prod_m (p_{i,t,c})^{o_{s,m}} * (1 - p_{i,t,c})^{1-o_{s,m}} \tag{6.1}$$

$$p_{i,t,c} = \frac{n^{(1)}_{i,t,c}}{n^{(1)}_{i,t,c} + n^{(2)}_{i,t,c}} \tag{6.2}$$

6.1.3 Model Predictions

Once we have computed the model parameters $p_{k,t,c}$ for each of the profiles B_k, in order to make a prediction we still need to be able to estimate to which profile each individual student belongs. For this estimation, we first compute a prior distribution of the bias profiles. Using this distribution the model can compute the expected answer of the student. Equation 6.3 shows how the probabilities of the different profiles are aggregated for an individual student s using the probability $P(s \in B_k)$ of belonging to the profile B_k in order to compute a personalised prediction $p_{s,t,c}$.

$$p_{s,t,c} = \sum_k p_{k,t,c} * P(s \in B_k) \tag{6.3}$$

Tracing Student's Bias profiles

The prior distribution is updated based on the sequence of observations of a given student. The probabilities $P(s \in B_k)$ are updated using Bayes Rule. Equation 6.4 shows how the model updates the expected distribution over the K bias profiles (B_k for k from 1 to K) based on observations (O_i) using Bayes rule. The Bayesian update from the observations allows subsequent predictions to be more precise because more observations for the student's profile are available.

$$P(s \in B_k | o_{s,m}, o_{s,m-1}, \ldots) \sim P(o_{s,m} | s \in B_k) * P(s \in B_k | o_{s,m-1}, \ldots) \tag{6.4}$$

6.1.4 Generalisation of Bias Profiles

We consider several versions of the Bayesian student model of inductive reasoning defined previously based on subsets of the student data. Given several independent subsets of the training dataset, the model can either assume that students exhibit a unique Bias profile over the whole dataset or exhibit different Bias profiles over the different subsets. By splitting the dataset in this way, the model cannot use correlations between parts of the data that are separated by the subsets. This will lead to lower performance in the case such correlations exist, but higher performance if such correlations do not generalise well. Without splitting the dataset, observing a student's answers to a given topic and comparison informs the model about expected answers on other topics and comparisons. Specifically, we use four conditions: not splitting (NONE), splitting by topics (TOPICS), by comparison (COMPARISON) and by topic and comparison (ALL).

6.2 Performance of the Model

We applied our model in the four different conditions described in Section 6.1. We compare the predictive capabilities of all versions and of a baseline model for the prediction of students' inductive reasoning. For the baseline, we estimate directly the expected proportion of each answer for a given question from the training data and make predictions on new data using this estimation as the expected probability. This baseline makes the optimal prediction for each question taken separately from the rest of the dataset but does not consider correlations between different questions in the students' sequences of answers.

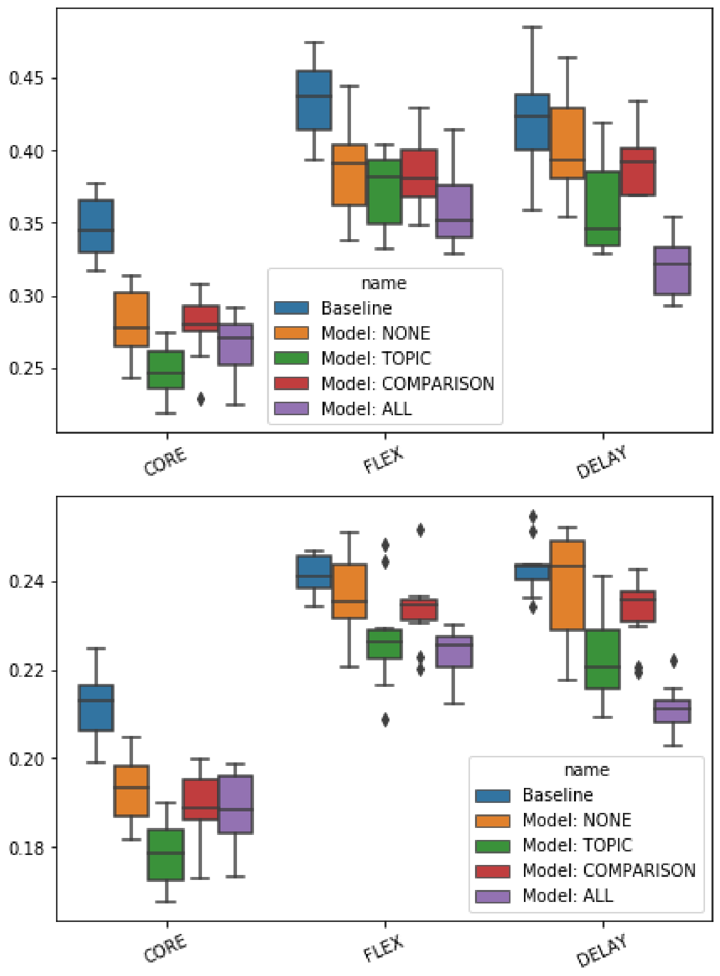


Figure 6.3 – Comparison of the model performance in terms of percentage of prediction errors (top) and mean square error (bottom) based on 10-fold cross validation.

Section	Core		Flex		Delay	
Model	ACC	MSE	ACC	MSE	ACC	MSE
Baseline	.654	.212	.566	.241	.580	.243
None	.721	.193	.610	.237	.597	.238
Topic	**.753**	**.178**	.626	.227	.639	.222
Comparison	.722	.189	.617	.234	.609	.233
All	.735	.188	**.641**	**.224**	**.681**	**.211**

Table 6.1 – Model performance in terms of Accuracy (ACC) and Mean Square Error (MSE) for the three types of questions: Core, questions the bias test section; Flex, questions following directly after adversarial feedback questions; Delay, questions delayed after adversarial feedback;

We evaluated the performance of our models both in terms of mean square error and accuracy. Figure 6.3 shows the distribution of the error rates for each of the models on a 10-fold Cross Validation and Table 6.1 reports the average. In each of the conditions, our model performed above the baseline model with approximately a 10% improvement on both accuracy error and mean square error. The model using the division in Topics performed best on the core questions as we could have expected given the correlations given in Chapter 5.

6.2.1 Modeling as a Tool to Analyse Student Behaviour

We have seen in the previous chapter that many correlations arise between the different feature comparisons for each topic. In particular, it is not directly obvious why some feature comparisons are correlated while others are not. For example, we can note that for the topic Geometry, the two questions of the bias test section on the comparison between the features color and type are correlated, which is an expected result as both questions are built on the same feature comparison. However, a more surprising result is that for the same topic, the answers to the comparison on the feature color and type is correlated with the answer to the comparison on the feature color and orientation but not correlated with the comparison on the features orientation and type (see Table 6.2).

Looking more closely at the data which generated these correlations, we are able to explain that this result comes from the fact that the feature color plays an asymmetrical role among the features and most students either use this feature as the most important or as the least important, but rarely do student place the feature color as the second most important between orientation and type. This explains why knowing the comparison between Color and Type tells a lot about the comparison between Color and Orientation.

A strength of our modeling technique is that the analysis can be read on the probability parameters of the extracted profiles B_k. For example, Table 6.3 shows the probability distributions for the comparisons C/O, C/T and O/T of four bias profiles extracted on the topic of geometry (with parameter $K = 4$). We can see in the profiles that high values of probability in the

Features	C	T
C	122	27
O	18	55

Features	C	T
T	100	64
O	40	18

Table 6.2 – Crosstables of answers selected for the questions on the topic of geometry. Counts of students on feature comparison C/T and C/O (left) and feature comparison C/T and T/O (right)

Comparison	C/O	C/T	O/T
Profile 1	0.74	0.76	0.94
Profile 2	0.85	0.76	0.48
Profile 3	0.18	0.27	0.91
Profile 4	0.50	0.29	0.16

Table 6.3 – Probability of observations for the three feature comparisons of topic Geometry for four Bias Profile automatically extracted by our model training algorithm.

comparison C/O (student is biased for color against orientation) are paired with high values in the comparison C/T (student is biased for color against type) and low values in the comparison C/O are paired with low value in the column C/T. But the column O/T contains high and low probabilities independently of the probabilities in the columns C/O and C/T. As we have shown with this example, we believe that analysing and interpreting a small number of extracted biases can give insight into the different profiles of students in a class. This analysis could be more efficient than looking through the data without specific preprocessing.

6.2.2 Flexibility of Students' Biases

In Chapter 5, we briefly analysed students' changes in biases caused by feedback. We found that our proposed student model allows us to have better insights into the students' changes in inductive biases.

Our model extracts a fixed number of bias profiles that cluster the students into several groups. Additionally, the model computes updated probability distributions of students' biases for questions following directly the adversarial feedback and for questions delayed after the adversarial feedback. These updated feature biases allow us to estimate how much each cluster of students changed. To analyse the flexibility of the different student profiles a common measure for changes in probability distributions is the Kullback-Leibler divergence (KL). The higher the KL estimate is, the more students have changed based on the adversarial feedback. Our hypothesis was that we would observe different levels of flexibility in the group of students. Figure 6.4 shows an example with $K = 10$ extracted bias profiles. We can observe that the profiles cover a large range of average KL divergence with more than 300% difference between the least and most divergent profiles. This confirms our hypotheses and shows that indeed some groups of students seem more susceptible to change feature biases based on negative feedback.

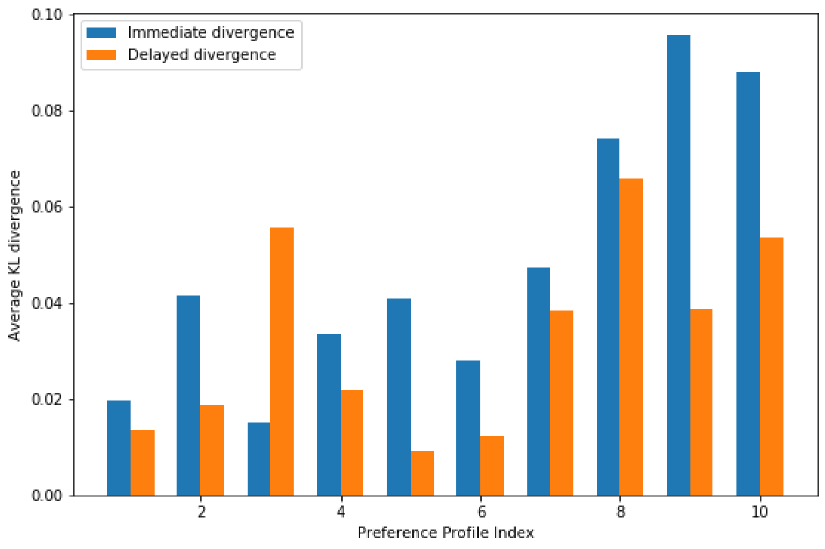


Figure 6.4 – Example of KL divergence for 10 extracted bias profiles

Furthermore, we hypothesised that changes would be stronger immediately after adversarial feedback and that some students will return to their original bias after a time delay. This is also confirmed by the evaluation of KL divergence displayed on Figure 6.4. We found that the majority of extracted profiles (70% in average with a standard deviation of 10%) have a lower KL divergence for the biases of the delayed question than for the biases for the flexibility question. This indicates that after the delay, the students partially go back to their initial bias.

6.3 Conclusion

In this chapter, we proposed a modeling approach for predicting and tracing students' biases during an example categorisation task. Our models are based on estimating biases as a probability measure and can further estimate how students change their behaviour based on receiving negative feedback. This work was motivated by the lack of inductive learning models able to describe individual differences of students and their change over time.

Our model performed well with an improvement of accuracy from 65% for the baseline to 75% for our model. This indicates that students indeed reason inductively using individual feature biases. Our model also revealed that students react to feedback differently and that they partially go back to their original biases after a delay. Both the lack of significant correlations on similar features across topics and the fact that splitting the dataset by topic led to higher predictive accuracy for our model showed that feature biases do not generalize well across topics. The lack of correlation in the students' inductive biases across the topics means that training the model on a given dataset will not generalize well to another dataset on a different topic. This lack of correlations could have been expected if we, for example, think about domains of expertise. It is natural to expect no clear correlations between understanding of theoretical physics, painting art, and running marathons. Different approaches to find generalizable features would be beneficial for using the kind of model that we described in practice.

Implications for Practice

The first advantage of a model such as the one we describe in this work is to be used in ITS. In this context models are used to predict and evaluate student mastery for skills and their learning process. This can then allow for adaptivity in the ITS [7]. Our model allows for ITS to use inductive teaching methods such as example categorisation tasks.

A second interest of our model is to be used to simulate students. Student simulations can be used to support the learning process in several ways. Firstly, this can help teachers design content and inductive teaching policies for learning environments [146, 147]. Simulations of students can also be used to develop an environment where teachers have to select examples to teach simulated students. Such an environment will allow teachers to analyse different teaching strategies, to extract optimal sets of teaching examples for inductive example categorisation tasks, or to practice teaching using inductive methods. For working in this direction we have already started to implement an application in which teachers choose examples in order to teach simulated students.

An additional interest in the model we proposed is that the model can give advanced insights about the students beyond what can be known from a direct statistical approach. Using this model teachers will be able to know for each student an assessment of their inductive bias and their flexibility. For example, this model could be used as a real-time learning analytics tool in the classroom during inductive teaching activities. Based on such information the teacher will be more able to decide which teaching interventions to use with each student. Our model allows for these advanced insights to be estimated in real time during activities and can for example be displayed on a dashboard for the teacher.

In conclusion, we contributed a probabilistic Bayesian model of students' reasoning processes for categorisation of examples in the context of an inductive learning activity. Our modeling approach is novel in two ways. First, it includes individual differences between students. We showed that students exhibit inductive biases and that the model we propose learns and predicts these biases using several bias profiles to describe students' behaviour. Second, the model accounts for changes in students' reasoning due to feedback. Using data from an inductive example categorisation task given to 222 students we showed that students have different biases which determine how they choose to answer ambiguous categorisation questions. These biases depend on the student preferred choice of features. Furthermore, giving the students negative feedback changes their bias but different groups of students have different levels of flexibility in that context. Additionally, we showed that students tend to go back to their original biases after a short delay. These aspects are part of the Bayesian model we proposed in this work and are a novel contribution from previous modeling approaches.

We believe that these two aspects are highly beneficial for using models of inductive reasoning in educational contexts.

7 Self-Improvement and Adaptivity for Inductive Teaching

In chapters 5 and 6 we motivated the use of student models of inductive reasoning. Yet, our model allows us only to trace the student biases and predict their answers but does not provide algorithms to directly compute optimal teaching strategies. Such algorithms are necessary for building adaptive and self-improving learning environments. In this chapter, we fill this gap and complete our theoretical foundations for learning environments supporting inductive teaching. In particular, we focus on providing algorithmic solutions to answer three research questions. **RQ1**: Given a set of examples, a target concept, and a student, how can examples be selected optimally to support the student's inductive reasoning process? **RQ2**: How does a learning environment adapt to students' individual differences during inductive reasoning tasks? **RQ3**: What are long term teaching strategies that allow an inductive learning environment to self-improve? These three research questions reveal challenges at three different levels of the capabilities of a learning environment. The first question assumes that the bias of the student is known. In this case, the learning environment must only aim to provide optimal teaching. For the second question, the student bias is not known. The learning environment must gain information about the student while trying to provide optimal teaching. Finally, the third question relates to the concept of self-improvement, which, as explained in Chapter 4, consists in gaining information from each interaction with successive students in order to improve over time. In this chapter, we provide formal definitions of the three research questions and their algorithmic solutions.

The student model used in this chapter is the one described previously by Tenenbaum to explain induction capabilities of human learners [231]. Students rely on their prior knowledge and biases to infer categories from examples using Bayesian inference. We mentioned this modeling approach in Chapter 5 and we define it mathematically in Section 7.1. This model can seem restricted to the narrow task of generalising from a set of examples to a category, but the task of using one's prior knowledge with observations to infer unknown information can be applied to many different contexts (notably, for most constructivist approaches to teaching).

Given a mathematical description of a learner, computing optimal teaching strategies is the task of interest in the field of Machine Teaching (MT). MT is an inverse problem from Machine

Learning (ML). In ML, the goal is to estimate the parameters $\hat{\theta}$ of a model as close as possible to the true parameters θ^* given an observed dataset $\mathcal{D}$. Instead, MT is the task of building a dataset $\mathcal{D}$ that would allow a model to estimate parameters $\hat{\theta}$ as close as possible to the true parameters θ^*. MT has been recommended multiple times as a way to improve education [260, 261]. However, a large part of the work done in this field makes unrealistic assumptions about the interaction between the teacher and the learner. In particular, the assumption is often made that the teacher has perfect knowledge of the student. Citing Zhu [259]: "[The teacher] *is almost omnipotent: it knows the world* θ^**, the learner's hypothesis space* Θ*, and importantly how the learner learns given any training data*". In practice, these assumptions do not hold. Students have different prior knowledge and exhibit different learning and reasoning mechanisms. Approaches try to alleviate this assumption and consider black-box models of students [137, 58]. In these models the teacher does not have perfect knowledge of the students, but can assess the students' knowledge through interactions. The black-box framework is more realistic in practice. This is why a lot of effort in understanding students' learning has been focused on inferring students' knowledge based on observation of their behavior and performance, often using Bayesian models (see Chapter 3).

In Section 7.1, we start by defining the learner model. Specifically, we provide a more detailed definition as to how students' reasoning about categories is influenced by observing examples. In Section 7.2, we apply the framework of MT and give an algorithm to select teaching examples. Additionally, we analyse several aspects of the optimal teaching strategies. In Section 7.3, we explore how the inductive learning environment can adapt to students, using several algorithmic methods. In Section 7.4, we describe how the algorithm proposed in Chapter 4 can be used in the inductive learning environment in order to self-improve.

7.1 Model of Inductive Reasoning

7.1.1 Context

A particular task in inductive reasoning consists of generalising categories from examples. Typically, a student would be given a set of objects among which some have been labelled as belonging to an unknown category (examples) or not belonging to the category (counterexamples). The students are then tasked to find a good explanation for the unknown category and generalise it to unlabelled objects. This sort of task will require the student to identify specific features of the presented examples and to discover new concepts or simple logical rules. How students are able to do this type of inductive reasoning is a challenging question and has been a topic of research for many years. One of the main approaches to solving this question has been to compare student's induction with some form of Bayesian inference [231].

The example on Figure 7.1 illustrates well the context of inductive reasoning. The figure shows 16 animals and three of them have been highlighted as being part of a category. Can you tell which of the other thirteen animals belong to the same category as the three highlighted

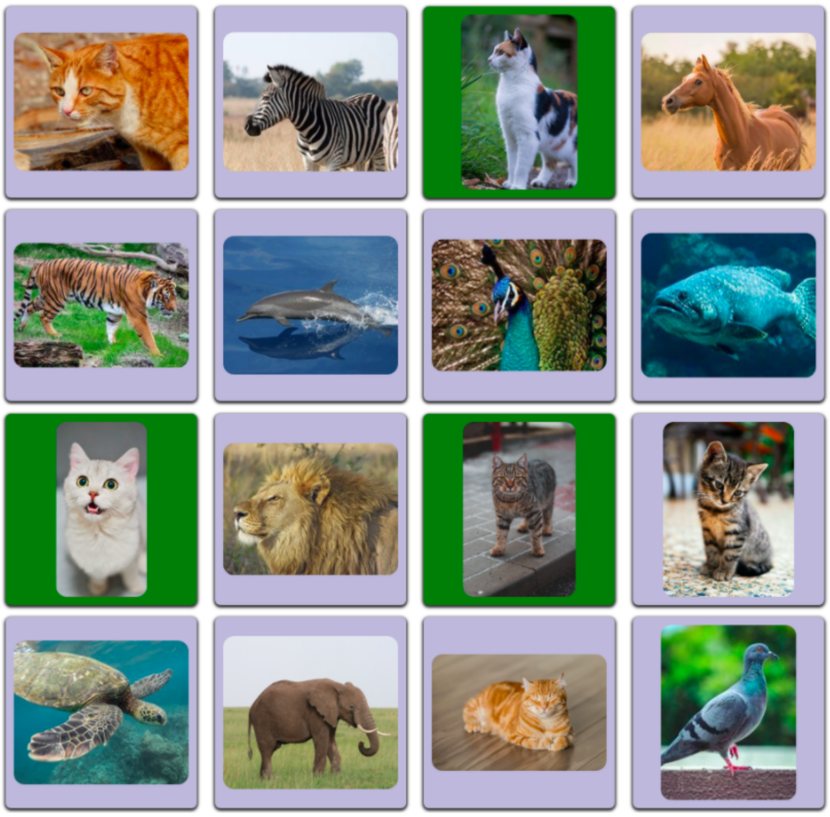


Figure 7.1 – Example of inductive reasoning about categories. Three examples of a category have been shown.

examples? From a purely information theoretic point of view, this question is impossible to answer. Multiple categories are compatible with the three highlighted examples, such as: "animals with fur", "mammals", or "felines". Thus, a purely deductive reasoning method cannot provide an indisputable answer. However, using an inductive reasoning approach, it seems very likely that the intended category is "cats". The Bayesian models of induction explain how this induction is done based on two main key concepts: *Prior Knowledge*: before seeing the highlighted examples, the student already has prior knowledge about categories. Categories such as "animal with fur", "mammals", or "felines" would, in general, be considered likely categories for the student. On the other hand, categories such as "animals that are either grey or can fly", or "cats and fishes" are unnatural categories and will be considered much more unlikely for most students. *Likelihood*: The examples given represent well some categories and are incompatible with other categories. Observing three cats as examples allows the student to be sure that the unknown category is not "elephants". These are also very likely examples for the category "cats" and somehow unlikely (but not impossible) for the category of "mammals".

Both prior knowledge and likelihood of the observed examples guide the students' answers in categorisations tasks. In this model, it is expected that students will guess categories that are both likely and well represented by the examples. These two aspects are the two probabilities on the right side of Equation 7.1. In the equation, $P(Category)$ is the prior knowledge about categories. The conditional probability $P(Examples \mid Category)$ is the estimation of how likely are some examples given a particular category. The posterior probability $P(Category \mid Examples)$ is how likely a given category is to be the correct answer after seeing the examples according to the student's reasoning.

$$P(Category \mid Examples) \sim P(Category) \times P(Examples \mid Category) \tag{7.1}$$

7.1.2 Mathematical Definition

The students are able to observe a collection of objects $\mathscr{O}$. Based on these objects we consider a set of categories $\mathscr{C}$ that can either contain all logically possible categories or a subset of the most likely categories. For a specific category $c \in \mathscr{C}$, each object $x \in \mathscr{O}$ is either classified as an example for this category or a counterexample if it does not belong to the category. A category c associates a label $y = c(x)$, which can take the values $+1$ (example) and -1 (counterexample), with every object x. During the task, some objects are labelled and some are not. (For example, on Figure 7.1 three examples are labelled and no counterexample is labelled). We will use the notation $d = (x, y)$ for a labelled object and the set $\mathscr{D}$ for the set of all examples and counterexamples shown to the student.

The student s has a prior $P_s(c)$ about each category, which corresponds to what we defined as the inductive bias in the Chapter 6. This prior can be thought of as what the student is likely to answer if the task is to pick a category before being given any labelled examples. Some categories such as "cats" or "elephants" will have a non-negligible probability, but most categories will be assigned a negligible probability, simply due to the exponential number of possible categories. Additionally, the student's prior is in general not arbitrary but originates from feature similarities between the objects as analysed in Chapter 5. When guessing the unknown category, the student assumes that the examples (respectively, counterexamples) are drawn uniformly from the set of examples (respectively, counterexamples) [231] and that whether they are shown an example or a counterexample is independent of the unknown category. This leads to computing the probabilities of observing a particular datapoint $d = (x, y)$ for a given category c according to Equation 7.2. In the equation N_c^- is the number of counterexamples and N_c^+ the number of examples. This probability is 0 if the example or counterexample given is incompatible with the category (for example if a cat is given as a counterexample then the unknown category cannot be the category of mammals).

$$P(d|c,y) = P((x,y)|c,y) = \begin{cases} \frac{1}{N_c^+} & \text{if } c(x) = y \text{ and } y = +1 \\ \frac{1}{N_c^-} & \text{if } c(x) = y \text{ and } y = -1 \\ 0 & \text{if } c(x) \neq y \end{cases} \tag{7.2}$$

Finally, students' bias about the different categories is influenced by the examples following Equation 7.1. With our new notation, the equation becomes: $P_s(c \mid \mathscr{D}) \sim P_s(c) \prod_{d \in \mathscr{D}} P_s(d|c)$.

7.2 Optimal Inductive Teaching with Knowledge of Students' Biases

In this section, we analyse the optimal inductive teaching strategy in the context where the learning environment knows the category c^* and the inductive bias of the student $P_s(c)$. The goal of the learning environment is to build the set of examples and counterexamples $\mathscr{D}$ such that the category c^* becomes more likely than every other category for the student. This condition is expressed by Equation 7.3.

For this type of optimisation, the MT framework also includes a measure of the cost of teaching. This cost can originate from the difficulty of generating examples or from the time required by students to analyse and compare multiple examples. Most commonly, the teaching cost is proportional to the size of the training dataset [259, 257]. In our case, we define $cost(\mathscr{D}) = |\mathscr{D}|$. Without limiting the number of examples, the optimal teaching strategy would be trivial and would consist only in selecting all objects in $\mathscr{O}$ as the teaching set $\mathscr{D}$. The task of the learning environment is to reach the teaching goal (Equation 7.3) while minimising the number of labelled examples.

$$\forall c \in \mathscr{C}, P_s(c^*|\mathscr{D}) \geq P_s(c|\mathscr{D}) \tag{7.3}$$

7.2.1 Formulation as an Integer Programming Problem

Below, we show that computing a set of examples and counterexamples satisfying the condition expressed in Equation 7.3 and minimising the cost $|\mathscr{D}|$ can be formulated as an Integer Programming (IP) problem. IP is a particular type of optimisation problem that is often used for practical optimisation problems such as scheduling or resource allocation or in the contexts of graph theory and number theory [53]. IP problems are, in general, difficult to solve. Yet, several exact and heuristic solvers are available and can prove useful in practice [109]. For example, within an educational context, IP has been used for optimising course timetables [161], allocating students to projects [8], or for college admissions [2]. All these applications used available solvers, justifying our IP formulation for optimal inductive teaching.

$$\begin{cases} \min_x \sum_i x_i \\ \forall j, \sum_i x_i D_{i,j} \geq L_j \\ x \in \{0,1\}^{|\mathcal{D}|} \end{cases} \tag{7.4}$$

The IP formulation (Equation 7.4) is derived as shown below. We use the following notations: $x_i = 1$ if the i-th example is selected in the teaching dataset $\mathcal{D}$, $x_i = 0$ otherwise; $D_{i,j} = log(\frac{P_s((x_i,c^*(x_i))|c^*)}{P_s((x_i,c^*(x_i))|c_j)})$ corresponds to the influence of example i on the likelihood between the category c_j and c^* and $L_j = log(\frac{P_s(c_j)}{P_s(c^*)})$ corresponds to the prior ratio of probability between the category c_j and c^*. Some of the terms $D_{i,j}$ can take infinite values when the example $c^*(x_i) \neq c_j(x_i)$. This occurs when the example or counterexample provided directly contradicts a possible category (for example, an example of "cat" contradicts the category "elephants"). In that case, it is equivalent to replace this parameter with a value sufficiently large, such that $D_{i,j} > L_j - n\delta$, where δ is the most negative value among the other parameters in the inequality.

$$\begin{aligned}
& P_s(c^*|\mathcal{D}) \geq P_s(c_j|\mathcal{D}) \\
\Leftrightarrow\ & P_s(c^*) \prod_{d\in\mathcal{D}} P_s(d|c^*) \geq P_s(c_j) \prod_{d\in\mathcal{D}} P_s(d|c_j) \\
\Leftrightarrow\ & \prod_{d\in\mathcal{D}} \frac{P_s(d|c^*)}{P_s(d|c_j)} \geq \frac{P_s(c_j)}{P_s(c^*)} \\
\Leftrightarrow\ & \sum_{i,x_i=1} log(\frac{P_s[(x_i,c^*(x_i))|c^*]}{P_s[(x_i,c^*(x_i))|c_j]}) \geq log(\frac{P_s(c_j)}{P_s(c^*)}) \\
\Leftrightarrow\ & \sum_i x_i D_{i,j} \geq L_j
\end{aligned}$$

7.2.2 Computational intractability

The IP formulation above has theoretical limits. The first limit comes from the exponential number of logically possible categories that could be considered. If we only restrict the problem to categories that are not identical on the set of objects $\mathcal{O}$ (for any two categories c_1 and c_2 there exist at least one object $o \in \mathcal{O}$ such that $c_1(o) \neq c_2(o)$), the number of categories to consider is still in the order of $2^{|\mathcal{O}|}$, which will render the optimisation computationally intractable. Additionally, when considering all logically possible categories, the majority of the categories will require more than half of the examples in $\mathcal{D}$ to be labelled. That is counter-productive as we chose to minimise the number of examples used for teaching.

A second limitation comes from the NP-hardness of the problem. Even if we consider examples with a smaller number of categories (not exponential, but polynomial in the number of objects), we can prove that the optimal inductive teaching problem is in the class of NP-hard problems. This implies that there is no known efficient algorithm to compute its solution [28].

The proof unfolds as follows. First, consider an instance of the Boolean satisfiability problem (SAT[1]). It consists of n Boolean variables $y_1, \dots, y_n$ which form $2n$ literals (positive and negative) and a conjunction of k clauses on these literals. To prove the NP-hardness of the optimal inductive teaching problem, we build an instance of this problem that allows us to solve the NP-hard SAT problem. Consider the inductive teaching optimisation problem with $2n$ objects corresponding to the $2n$ literals; n categories, one for each variable, which have for only examples the two objects associated to the literals of this variable; k categories, one for each clause, which have for examples the objects associated to the literals of that clause; c^* to be the empty category which has only counterexamples; $P_s(\cdot)$ chosen such that $\forall c \neq c^*, P_s(c^*) < (\frac{1}{2n})^n P_s(c)$. It can be verified that the optimal inductive teaching problem will have a solution with exactly $|\mathscr{D}| = n$ if and only if the SAT problem is solvable. Furthermore, an optimal inductive teaching dataset directly determines an assignment of literals to solve the SAT problem. We have thus shown that the optimal inductive teaching problem is at least as hard as SAT in the general case, thus proving the NP-hardness.

We have proven that, in the general case, there is no known efficient algorithm to select an optimal set of examples and counterexamples for inductive teaching. Given this limitation, there remain three options: using inefficient exact optimisation, but restricting the use cases to small cases (with about 5-10 categories and objects), using approximate optimisation for which more efficient heuristic are available, or making additional assumption on the structure of the considered categories and of students' reasoning to allow for more efficient computation of an optimal teaching set. If such an additional assumption does accurately describe students' reasoning, then, this third option would lead to both efficient and optimal inductive teaching.

7.2.3 Optimal Teaching for Categories Built on Objects' Features

The intractability in the general case leads us to consider more practical cases where only a limited number of categories can be imagined by students. Following our analysis from Chapter 5, the students' inductive biases can be attributed to the objects' features. Given a small number of features $f_1, \dots, f_k$, we consider that the students construct categories from each of the features, and from conjunctions of the features and their negations. For example, if "white" and "flying" are two features, a student would consider a number of categories: "white animals", "flying animals", "non-white animals", "non-flying animals", "flying animals that are not white", "white animals that fly", "white animals that do not fly", and "animals that do not fly and are not white". In this context, for a category c, each feature f_i either must be positive, or must be negative, or is unimportant.

[1]https://en.wikipedia.org/wiki/Boolean_satisfiability_problem

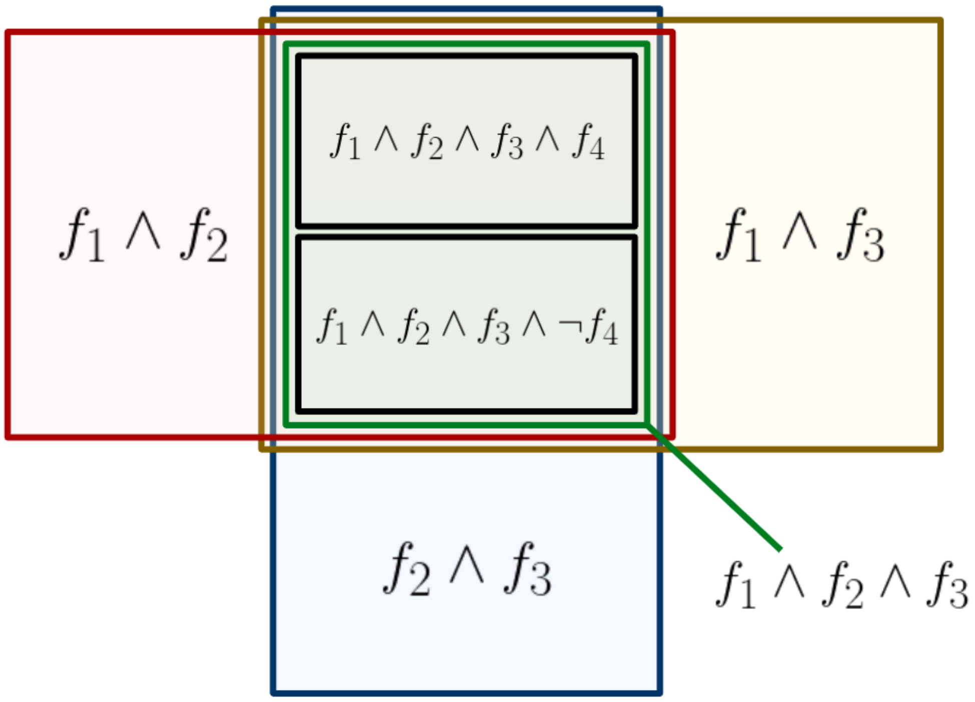

$f_1 \wedge f_2 \wedge f_3 \wedge f_4$
$f_1 \wedge f_2 \wedge f_3 \wedge \neg f_4$
$f_1 \wedge f_2$
$f_1 \wedge f_3$
$f_2 \wedge f_3$
$f_1 \wedge f_2 \wedge f_3$

Figure 7.2 – Venn Diagram illustrating the structure of students' inductive bias, when being restricted to categories built on conjunction of features.

An efficient teaching set can easily be computed in this case. Considering the target category c^* that is constructed as a conjunction of the m features $f_{i_1}, \ldots, f_{i_m}$, an efficient set of examples and counterexamples can be computed as follows.

First, all categories defined on other features than $f_{i_1}, \ldots, f_{i_m}$ can be rejected by using only two examples. It is enough to set the two examples to have the proper fixed values for the features $f_{i_1}, \ldots, f_{i_m}$ and have opposite values on all other features. These two examples would be added to the training only if they are required. It is the case when categories using other features are more likely than c^* according to the student's inductive bias.

Second, the only remaining categories are categories that are constructed from a subset of the features $f_{i_1}, \ldots, f_{i_m}$. Each of these categories c, can be eliminated with one of two ways: either providing additional examples inside of c^* (a category defined using a subset of the features defining c^* is necessarily larger because it has fewer constraints) or providing one of the m counterexamples that have exactly one missing feature. The optimal set of examples in that case can easily be computed using a computationally tractable algorithm. The teaching dataset obtained in this way will be at most two examples more than the optimal dataset. An additional assumption could be that students consider equally all the categories formed as described above. In that case only the two examples from the first step will be necessary.

An illustration of the structure of students' inductive bias under the simplifying assumption is

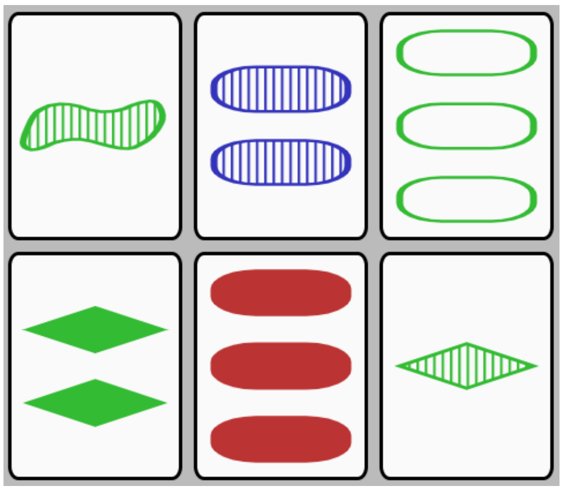


Figure 7.3 – Example of cards from the card game Set

shown on Figure 7.2. The target category is the category $f_1 \land f_2 \land f_3$ represented in green. The first step of the teaching strategy that we propose consists of eliminating the many categories that are defined using features other than the features of the target category (two examples are represented in black color on the figure). The other categories all contain the target category c^* (three examples are shown in red, blue and yellow on the figure). We propose to use an optimised combination of examples and counterexamples. With three features to define c^* only three counterexamples are enough to eliminate all other categories than c^*.

Example of Complex Category

The examples mentioned above are limited to conjunctions of a subset of the features or their negations. Other logical formulas could be used to combine the features and form arbitrarily complex categories. For example the category "white or fly animals, but not white flying animals" is logically constructed from the two features. This category contains white animals that do not fly and non-white animals that fly. It could be written with the logical symbol for the exclusive disjunction "white $\oplus$ flying". A similar example in geometry could be the categories of "rectangles and rhombuses, but excluding squares". This sort of category appears only in uncommon circumstances and it is natural that students will assign very low prior probabilities to them. If it is expected that students will not construct such complex categories, we recommend applying the example selection defined above. Otherwise, the learning environment should rely on the formulation given by Equation 7.4.

An example of such complex categories comes from the card game, Set[2]. Examples of cards from the game are shown on Figure 7.3. The cards can be described with four features, which can each take three values: the color C (red, green, or blue), the number of symbols N (1, 2, or 3), the shape S (round, rhombus, or squiggle), and the filling F (empty, stripes, or full). In this

[2]https://en.wikipedia.org/wiki/Set_(card_game)

game, the rules rely on a disjunctive logical formula, which, as argued previously, is a priori less likely than simpler conjunctions of features. The goal of the game is to find triplets of cards that are called Set and follow the rule, "If two cards have it and the third does not, then it is not a Set"[3]. To state the rules using a logical formula, we can first define $I(X)$ and $D(X)$ for triplets where the value of the feature X is respectively identical or all different among the three cards. With this notation, the rule of the game Set is to find triplets of card that verify the logical formula: $(I(C) \vee D(C)) \wedge (I(N) \vee D(N)) \wedge (I(S) \vee D(S)) \wedge (I(F) \vee D(F))$. It could also be written in a slightly simpler form: $\forall X \in \{C, N, S, F\}, I(X) \vee D(X)$.

The web interface shown in Figure 7.4 was used in a preliminary experiment on inductive reasoning (similar to the experiment described in Chapter 5). The goal was for the participants to induce the rule of Set from examples of correct and incorrect triplets. The interaction with the induction interface happened in a sequence of steps as follows. First, one correct triplet and one incorrect triplet are shown and the student must write any explanation of why the triplets are categorised in this way. At that stage, multiple explanations are possible and it is not expected that the students will find the correct explanation. At every following step, one correct triplet and one incorrect triplet are added to the list of examples, the student is then prompted to verify if the previously chosen explanation is still accurate, if not, the student must submit a new explanation. The sequence of steps finished after 10 examples and 10 counterexamples.

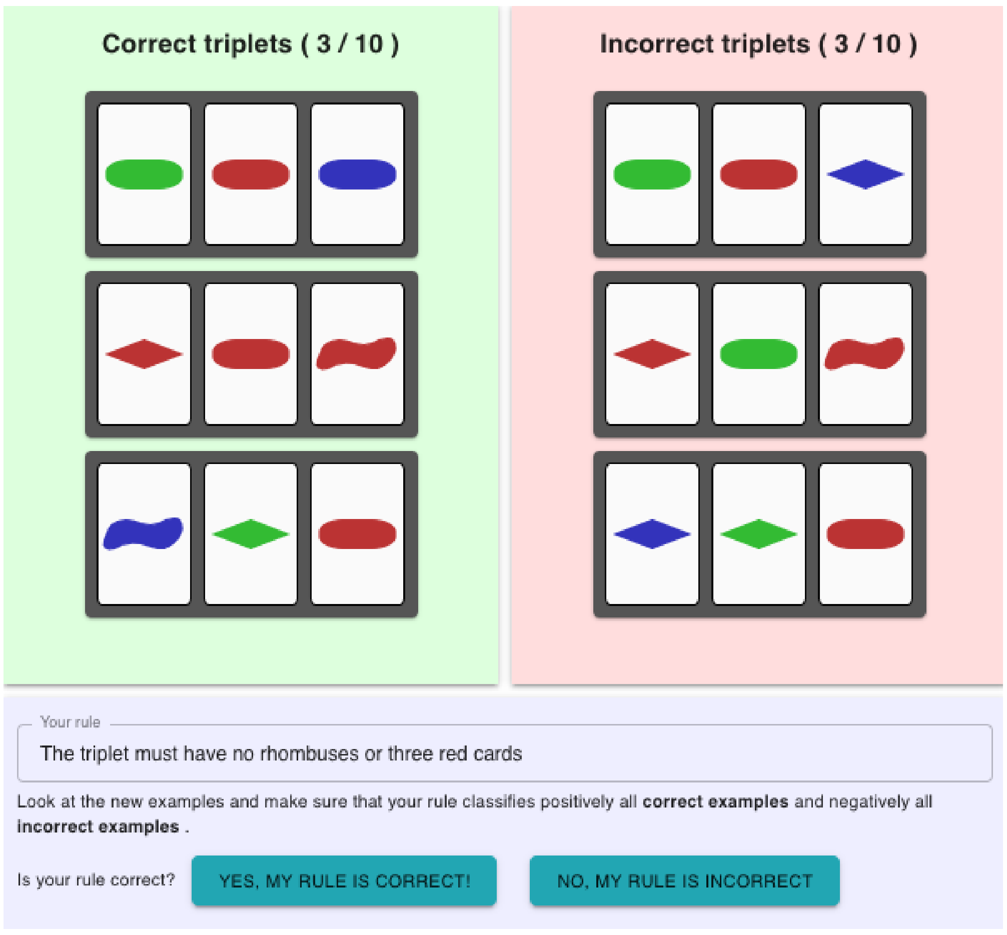
Our preliminary experiment indicated that different sets of examples influenced the ability of students to guess or not the correct categorisation rule. Additionally, we observed that some students who had a computer science background naturally used logical operators in their answers while others did not. This confirmed that students use their prior knowledge, which influences how they analyse and reason about examples and categories. Additionally, it confirmed that such complex logical rules will be given very unlikely probabilities in most students' inductive biases unless they are familiar with such types of categories.

7.2.4 Insight from the inductive teaching strategy

In the inductive reasoning model defined at the beginning of the chapter, given an example or a counterexample d, a student updates his belief using Bayes rule (see equation 7.1). $P(c^*|d) = \frac{P_s(d|c^*)P_s(c^*)}{P_s(d)}$ is maximal for an example that maximises the ratio $\frac{P_s(d|c^*)}{P_s(d)}$. This already gives several insights concerning the optimal selection of examples.

Firstly, if the category c^* has a large number of examples and few counterexamples, then it is, according to the learner model, more efficient to support the induction using counterexamples than examples. Indeed, the likelihood term $P(d|c^*)$ is lower for examples than for counterex-

[3]Personal note from the author: From my experience explaining the rule of this game to many friends and family members, stating the rule this way (although it is perfectly accurate) leads nearly always to complete incomprehension. So far, I have only been able to explain the rules by showing multiple examples and repeating the rules in different ways



The triplet must have no rhombuses or three red cards

Look at the new examples and make sure that your rule classifies positively all **correct examples** and negatively all **incorrect examples** .

Is your rule correct?

Figure 7.4 – Interface for learning the rules of Set inductively

amples in that case. Inversely, if the category c^* has few examples, it is better to use examples to support the inductive reasoning process. To illustrate this aspect, we can consider the example from Figure 7.1 for teaching the category of "cats". It can be very efficiently guessed from a few examples of cats. On the other hand, in order to make a student guess the category "mammals", it would seem more appropriate to explain it by showing counterexamples of "birds" and "fish", because twelve out of the sixteen animals are mammals.

A second insight comes from the term $P_s(d)$. The example or counterexample selected will be more effective if it is unexpected for the student. This is because data points that are more surprising influence the posterior probability more (see Equation 7.1). Another way to explain this is that an example is surprising if it contradicts a number of likely categories c that are not the target category c^*. Because $P_s(d) = \sum_c P_s(d|c)P_s(c)$, an example or counterexample has a lower probability $P_s(d)$ if it is incompatible ($P_s(d|c) = 0$) with some of the categories that the student considers likely (high $P_s(c)$). To teach optimally the learning environment should use examples that exclude the incorrect categories that the student would otherwise find most likely.

A third insight is that this probabilistic model of student inductive reasoning also explains near-miss examples. A near-miss is an example or counterexample that is very similar to the opposite category and usually differs in only one way. Such examples have been shown to be important to consider for learning concepts and categories [154, 253]. For example, a student might not know that dolphins are mammals. This student's inductive bias would thus give more weights to the categories "fish and dolphins" and "mammals without dolphins" than to the categories "fish" and "mammals". It should be noted that the student does not realise that the categories he or she thinks of as "fish" is the "fish and dolphins" category. This misconception of the student can be compensated within inductive teaching by selecting examples of dolphins along with examples of mammals and counterexamples of fish. Additionally, the efficient teaching strategy for the case of categories built on conjunctions of features uses counterexamples with exactly one missing feature, which confirms the previous results on near-miss examples.

7.3 Adaptivity

We have shown in Chapter 5 that students differ in their inductive biases. For this reason, the case described in the previous section, where the learning environment knows the bias of the student, is not directly usable in practice without a mechanism for estimating the biases of students. Following the same principle as the model we described in Chapter 6, the learning environment will start with a prior about the student biases, then, will better estimate the student biases over time by observing the student's answers and interaction with the learning activities. In this section, we first provide a strategy for teaching efficiently under uncertainty, then we analyse how the strategy can be changed to be adaptive by making use of observations of the students.

7.3.1 Batch Teaching Strategy with Uncertainty

The teaching strategy discussed in the previous section is called batch teaching, because the set of examples and counterexamples is shown all at once to the learner. A batch teaching approach is still possible with uncertainty about the student's inductive bias. If, for example, we consider any number K of inductive biases $P_k(c)$ for $k = 1, \ldots, K$. With $L_j^k = log(\frac{P_k(c_j)}{P_k(c^*)})$, we obtain the updated IP problem from Equation 7.5. The direct consequence of this new formulation is that all the constraints on the teaching goal are, in general, more restrictive than in the case of a singular inductive bias.

$$\begin{cases} \min_x \sum_i x_i \\ \forall j, \sum_i x_i D_{i,j} \geq \max_k L_j^k \\ x \in \{0,1\}^{|\mathcal{D}|} \end{cases} \tag{7.5}$$

This problem is equivalent to considering a single aggregated bias $P_a(c_j) = \frac{\exp(\max_k L_j^k)}{1+\sum_m \exp(\max_k L_m^k)}$ and $P_a(c^*) = \frac{1}{1+\sum_l \exp(\max_k L_l^k)}$. A dataset that reaches the teaching goal from Equation 7.3 for this aggregated bias will necessarily use at least as many examples and counterexamples as each of the inductive biases $P_k(c)$ for $k = 1, \ldots, K$. In particular, we observe that $\forall k,\ P_a(c^*) \leq P_k(c^*)$, which means that the category c^* is less likely according to the aggregated than according to every other student bias.

The batch teaching with uncertainty is useful when a system must find a unique set of examples that would be satisfactory to multiple students with individual differences in their inductive biases. This scenario is also similar to the case of selecting examples to show simultaneously to all the students in a classroom. For this case, the teacher would do best to select examples by solving the IP problem on Equation 7.5. The case of optimising the selection of examples to teach a large group of students simultaneously has been studied in the context of MT for a linear classifier learner [257]. In that work, the authors also show the importance of adaptivity. In particular, they proposed to divide the classroom into homogeneous groups in order to optimise the teaching sets for each group. Estimating the inductive bias of a student and adapting the teaching examples is our focus in the following subsection.

7.3.2 Optimal Adaptive Teaching Strategy

Observations of Student Answers

When a learning environment has uncertainty about the state or individual characteristics of the student, the only possibility to reduce this uncertainty is to observe the student. This observation can take the form of data collected before the student interacts with the learning environment (for example, a registration form) or data that the learning environment collects from the student interaction with the system (for example, answers to multiple-choice questions, or behaviour when solving problems).

Within MT, one of the frameworks for realistic interactions between a teacher and a student is Iterative Machine Teaching (IMT) [136, 257]. In this framework, the teacher does not interact with the student in one-shot but instead iteratively. Each iteration consists of two steps. First the learning environment provides teaching. Second, the learning environment queries information from the student. The information received about the learners varies between different IMT approaches. Often this information is unrealistic given a pedagogical scenario with real students. Yet, some approaches assume limited information about the student state and learning processes and rely on actively querying the student for assessing its knowledge [137]. We will adopt this angle throughout the rest of this chapter.

In the case of the inductive teaching scenario that we analyse in this chapter, we specifically focus on the answers given by students to categorisations questions. Additionally, as we indicated previously, we model that the students answer probabilistically according to their posterior as defined by Equation 7.1. This modeling step is justified by our results given in Chapter 6.

Estimating Student's Prior

One of the main tasks for an adaptive system is to refine estimations about the state of the student as new data becomes available. Consider the case where the learning environment has uncertainty about the student's inductive bias and has discretised the possibilities into K possible biases. We use the notations B_i for $i = 1,\dots,K$, S_i for the fact that the student s has the prior B_i, $P(S_i)$ for the uncertainty of the learning environment about student s, and i^* the unknown index of the true student's inductive bias.

$$\forall i \neq i^*,\ \mathbb{E}[log(\frac{P(S_{i^*} \mid c_1,\dots,c_m)}{P(S_i \mid c_1,\dots,c_m)})] = log(\frac{P(S_{i^*})}{P(S_i)}) + m * D_{KL}(B_{i^*} \| B_i) \tag{7.6}$$

When the student is queried to answer with a chosen category, the student's answer is an indication of the student's inductive bias. In particular, observing the answer c happens with probability $P(c \mid B_{i^*})$. The observation allows the learning environment to update the estimation of the student's bias. Equation 7.6 gives the learning environment's expected uncertainty about the state of the student after making m observations. In the equation, $D_{KL}(\cdot\|\cdot)$ is the Kullback-Leibler divergence. We observe that the learning environment can get an arbitrarily good precision $P(S_i) < \delta$ for $i \neq i^*$ with a number of queries m, which will be in the order given by Equation 7.7. The number of queries grows only logarithmically in the desired precision, which means that only a few queries will be sufficient to reach satisfactory certainty about the student's inductive bias. Additionally, even if the learning environment at first assigns a very small probability to the correct bias $P(S_{i^*})$, this will also be compensated efficiently.

$$m \sim \frac{1}{D_{KL}(B_{i^*} \| B_i)} log(\frac{P(S_i)}{\delta \, P(S_{i^*})}) \tag{7.7}$$

Equation 7.7 also has a term that is the inverse of the KL divergence. This means that, if the KL divergence between two inductive biases is small, the learning environment will distinguish between the two biases less quickly. In that case, it is also expected that the two biases would have very similar optimal teaching strategies. To solve this problem, when clustering the students between biases, it is beneficial to ensure that these biases are sufficiently different to be used meaningfully by adaptive teaching strategies.

Partially Observable Markov Decision Processes

We have shown that the students' inductive biases can be efficiently estimated. From that point, a strategy could be as follows. First, query a student multiple times. Then, apply the optimal teaching strategy described in Section 7.2 once the uncertainty about the student's bias is sufficiently small. It is, however, possible to find even more efficient methods by allowing for the possibility to alternate or not the actions of showing examples and querying information about the students' inductive biases. The commonly used framework for this type of optimisation is Partially Observable Markov Decision Processes (POMDP). POMDP are a very common framework for optimisation and decision making under uncertainty, which has been used successfully in robotics [128], health-care [103], and education [190]. When applied to education technology, the POMDP framework is similar to the framework that we defined in Section 3.2.3. In this framework, students have a current state and can transition to different states, the learning environment can select pedagogical actions and students states cannot be observed directly but instead describe probability distribution of observation. POMDP have been used on learner models of the same kind as the one we analyse in this chapter to compute efficient teaching policies [190]. An algorithm to solve a POMDP would provide a policy that describes the optimised decision making mechanism. In this chapter, we often use the term teaching strategy to refer to the policy.

One of the main advantages of POMDP optimisation is that they allow balancing between actions that collect information about the student and actions targeted to teach. For the inductive teaching scenario, two actions can be considered. The first consists in showing a labelled example or counterexample to the student, $d = (x, y)$. The second consists in querying the student for guessing a category. An efficient strategy would balance between these two actions. The first one is necessary to guide the induction of the student, and the second one is important to learn the bias profile of the student in order to better optimise the choice of examples in the first action. In practice, these actions will have an associated cost, such as the time required by students to analyse a new example or to answer a query. For example, it could require students one minute to perform the first action and only 20 seconds for the second. These costs will determine the optimal teaching strategy.

Because optimal policies for POMDP are, in general, computationally intractable, it is necessary to use online methods that produce an approximation of the optimal policies. In particular, we prove the intractability for the optimisation without uncertainty about the student's inductive bias, which is only a special case of the optimisation with uncertainty. Online approximations for POMDP with limited look-ahead have been shown to handle large POMDP efficiently [200]. We show below how to perform such optimisation. First, Equation 7.8 shows how to compute the probability that the student will guess the correct category $P(c^*|d)$ in order to select an example or counterexample d that maximises this probability. The second Equation 7.9, allows the learning environment to estimate the probability that the student will give the correct answer if it chooses to first gain information about the student before showing an example. In the equation, $P(c) = \sum_i P(B_i)P(c|B_i)$ is the probability of observing the answer c when querying the student for a category.

$$P(c^*|d) = \sum_i P(B_i)P(c^*|B_i, d) \tag{7.8}$$

$$P(c^*|\text{Observation First}) = \sum_c P(c) \max_d \sum_i P(B_i|c)P(c^*|B_i, d) \tag{7.9}$$

7.4 Self-Improvement

In the two previous sections, we provided algorithms to select an optimal set of teaching examples and counterexamples when the learning environment knows the inductive bias of each individual student. Furthermore, we showed how to adapt to students when the learning environment does not know with certainty the inductive bias of the student but is able to quantify its uncertainty about the students' inductive bias. In this section, we focus on the task of quantifying an initial uncertainty about the students' biases and optimising the selection of teaching examples when lacking information about the students' prior. To this end, we show how the algorithms defined in chapters 3, 4 and 6 can be adapted.

7.4.1 Multi-Armed Bandit Optimisation

The methods described in Section 7.3 rely on knowledge about the distribution of inductive biases within the population of students. As long as the learning environment does not have access to sufficient quantity of student data, the adaptive teaching strategy would face a sort

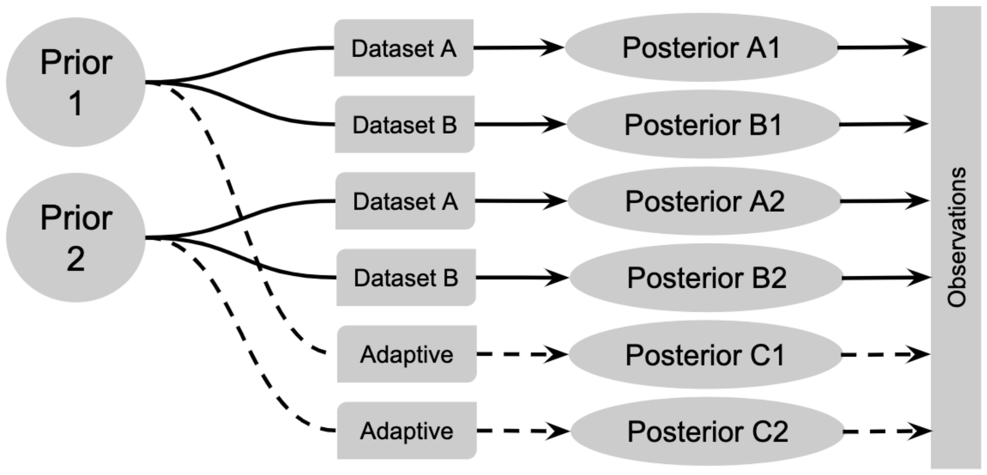


Figure 7.5 – Example of models for the inductive reasoning task

of cold start problem. For example, this can occur if a new topic is introduced in the learning environment. A solution to that problem is the optimisation algorithm based on Thompson Sampling that we described in Chapter 4. We remind the reader that MAB optimisation can be successfully used in the context of a cold start. The method can be applied directly to the case of initial data collection for the inductive teaching learning environment.

Figure 7.5 illustrates how the learning environment could apply MAB optimisation. The first step would consist of showing one of several inductive teaching datasets, then querying the students for guessing the category. The initial datasets can be chosen by expert educators or optimised using student models and simulations that do not require student data. Additionally, the MAB optimisation requires that the learning environment defines the utility function, which, in this case, consists in evaluating desired and undesired posterior beliefs.

The six posteriors depicted on Figure 7.5 do not necessarily need to be distinct. If we consider, for example, two students, Alice and Bob, such that Alice cares equally about cats and dogs, but Bob only cares about dogs. After being shown a counterexample that eliminates the category of cats, Alice and Bob will have the same posterior.

A difference with the algorithm we proposed in Chapter 4 is that here, we do not estimate the transition probabilities of the different teaching datasets. These transition probabilities from prior states to posterior states are known and given by our model in Section 7.1. Instead, the optimisation process maximises the probability for the students to reach desired posterior states and the collected data over time will allow us to estimate the distribution of prior states (inductive biases).

7.4.2 Extracting Students Inductive Biases

The adaptive teaching strategy provided in the previous section relies on the knowledge of a discrete set of student biases and how students are distributed among these biases. This allows the learning environment to adapt to students by estimating which of these biases students can be assigned.

Extracting such inductive biases can be done with the Expectation-Maximisation algorithm used in Chapter 3 and Chapter 6. Additionally, the algorithm forms clusters of students with similar inductive biases. In practice, the algorithm assumes a fixed number of latent inductive biases. Then, the algorithm estimates parameters for both the biases and the distribution of the students within the clusters.

The learning environment, after having collected a sufficient quantity of student data, can use the adaptive teaching strategy defined in Section 7.3. In order to ensure the quality of the adaptive strategy, the learning environment can integrate this adaptive strategy within the MAB optimisation process. The result of this would be that as long as the adaptive strategy does not perform better than the initial datasets, due to low amounts of data, it will not be selected as often.

7.5 Conclusion

In this chapter we contributed several algorithmic solutions to different aspects of the implementation of learning environments for inductive teaching. Although our solutions have not been tested empirically, they are optimal according to the model that we defined in the beginning of the chapter. This model has been tested empirically, and is also supported by our study reported in Chapter 6. The lack of empirical evidence is a limitation of our contribution.

First, we have analysed the case of the selection of optimal teaching examples when the student's inductive bias is known. We showed that this problem is equivalent to an IP problem. IP problems are in theory computationally intractable, but multiple solvers and heuristics are available and have been used successfully in an educational context. Additionally, simplifying assumptions about students' reasoning leads to the possibility to efficiently compute efficient teaching strategy. In particular, we analysed the case of categories constructed from conjunctions of features of the objects and their negations.

Secondly, we have analysed the optimal teaching when the learning environment has uncertainty about the inductive bias of the student. In that case we have shown that batch teaching is still possible and is equivalent to teaching a student with an inductive bias, which assigns less probability to the goal category c^*. Yet the algorithmic solution is the same as in the first case. Another more interesting approach for adaptivity is to consider the frameworks of IMT or POMDP. The learning environment makes observations about the student's knowledge at each step of the learning process. We have shown how this framework can be used for the inductive teaching scenario and provided the necessary steps to apply such optimisation. Additionally, we found that the students' biases can be efficiently estimated from asking the students to guess the correct category.

Finally, our third contribution is the algorithms for self-improvement. We showed two possibilities. First, the necessity to compute and refine a model of students' inductive biases as more student data is collected. This can be done with the Expectation-Maximisation algorithm that we already used in Chapter 6 to compute the unobserved parameters of students latent states. Second, we showed how to apply the self-improving Thompson Sampling methods described in Chapter 4.

This chapter completes the main contribution of the thesis by providing the necessary algorithms for students modeling and for computing efficient teaching strategies. Our contributions allow us to develop learning environments for inductive teaching that will be both adaptive and self-improving.

8 Assessment of Students' Uncertainty

Student assessment is necessary to be able to measure the effect of pedagogical interventions [17]. Whether it is for adaptivity or for self-improvement a learning environment necessarily requires making some observations of students in order to infer their current states and state transitions. Assessment consists in observations of students that are focused on estimating their state of knowledge or ability level. Assessment can be improved at two levels: first, the modeling and analysis of observations, which we have previously discussed in Chapter 3 and second, the data collection mechanism. In this chapter, which is the last of the thesis, we want to focus on this second possibility. Improving the data collection mechanism is complementary to our contributions from previous chapters. It is another key element of adaptive and self-improving learning environments that has the potential to greatly improve educational technology.

Assessment of students is at the core of student modeling in adaptive learning environments. Within the framework of this thesis given in the introduction, assessment is the mechanism to collect data from students and to infer students' states on which adaptive decisions will be based. Assessment can take many forms, such as quizzes, homework exercises, projects, essays, or in-class examinations. Additionally, we distinguish two types of assessments. Teachers use formative assessments in order to adapt the teaching to students and summative assessments in order to evaluate students after a learning sequence [71].

Research on student models improves the inference of students' knowledge states based on currently existing data. On the other hand, including improved mechanisms for collecting data from students is also a key aspect that deserves attention in the design of automated learning environments. It is useful to consider the case of a human tutor to better understand this idea. A human tutor, when asking a question to a student, does not only rely on the received answer. From the student's attitude, tone of voice, facial expression, or body gestures, the tutor can perceive whether the student is guessing the correct answer and how confident the student is. These are all important cues helping the tutor to better adapt to the student. ITS relying on BKT lack access to such cues and must handle the uncertainty about students' true knowledge states. As a consequence, the system regularly gives unnecessary exercises to a student because it was not able to be sure that the student had the correct level of mastery. Additionally, BKT has been used with a threshold of 95% probability of mastery [55]. It means that the ITS will wrongly validate the mastery of a skill for about 5% of the students while they have only been lucky in answering the tutor's questions. These errors can be tuned at the level of the student model but will also be reduced with more useful observations of the students. There is a lot of work within the Learning Analytics community aimed at enabling automated systems to capture more data from students by using numerous digital sensors [23]. This range of methods, however, loses in efficiency and scalability because they usually rely on additional hardware such as cameras, eye trackers or wearable devices.

In this chapter, we focus on probabilistic testing [195] and in particular, on the advantages it has as an alternative answering mechanism for Multiple-Choice Questions (MCQs). A probabilistic test, similarly to a MCQ, consists of a question and a list of possible answers among which is one correct answer. Probabilistic testing differs from MCQs, because students must assign a probability to each choice corresponding to how likely they think the choice may be the correct answer, instead of only selecting a unique answer. This mechanism still supports the scalability and efficiency of development and administration of MCQs while also supporting the estimation of the uncertainty of student answers. Estimating students' uncertainty may support more accurate predictions of student knowledge. In particular, this answering mechanism is complementary with the inductive teaching strategies that we defined and analysed in Chapter 7. When a student is guessing categories based on a set of examples, this new answering mechanism will allow the student to not only pick one answer but to provide multiple answers with probabilistic weights.

In Section 8.1, we discuss multiple alternative answering mechanisms for Multiple Choice Questions (MCQs) and focus on the advantages of reducing students' guessing and estimating students' confidence. In Section 8.2, we describe in detail the concepts of probabilistic testing and proper scoring rules and report some preliminary observation from a web application for probabilistic MCQs. In Section 8.3, we analyse the advantages that this type of assessment could bring to the optimisation of inductive teaching that is the central concern of the thesis.

8.1 Alternative Answering Mechanisms for MCQs

Within educational systems, student assessments play a major role in evaluating students' current knowledge. To support efficiency and scalability, often MCQs are used as an assessment tool. They consist of a question, a correct answer and one or several incorrect choices (distractors). Most often the student is shown the question and all of the choices (among which is the correct answer) and has to select the choice that he or she thinks is the most likely to be correct.

One of the many advantages of digital technology is that it enables the development of online assessment systems [11]. MCQs are particularly well suited for such online learning environments because of the ease in which questions can be distributed and graded automatically

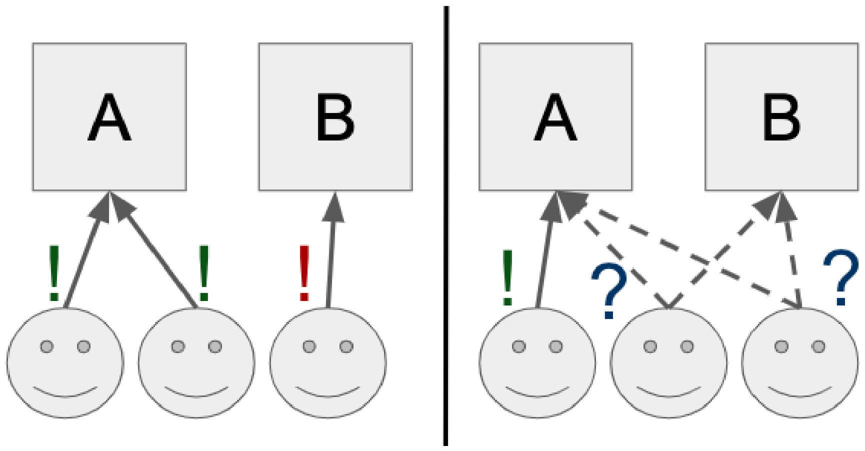


Figure 8.1 – Comparison of two cases: (left) Two thirds of the class know the correct answer A and one third wrongly thinks B is the correct answer; (right) One third of the class knows that A is the correct answer and two thirds of the class answer randomly between A and B.

[4, 33]. Furthermore, automated online assessment systems using MCQs are perceived positively both by students and by teachers [11, 35]. Although they are efficient in many aspects, MCQs have several limitations. In this section we discuss how alternative answering mechanisms for MCQs can reduce students' guessing, measure partial knowledge, and allow us to estimate students' confidence.

8.1.1 Reducing Guessing

MCQs can mask a student's true understanding since the questions rely on guessing between the likely answers if a student is unsure of an answer. Thus, it can be difficult to know if the student has learnt the required knowledge or if their answer was due to a random guess. For example, consider the two situations shown in Figure 8.1. In a classroom, if the teacher observes two thirds correct answers and one third wrong answers, the teacher cannot know whether this observation comes from two thirds of the class answering randomly between A and B or comes from one third of the class wrongly answering B with high confidence or another possibility. The issue arising is that the teacher will observe the same data (answers to the MCQ) based on very different student knowledge in these two very different scenarios. In order to support teachers, it is very desirable to identify such differences in students' knowledge states. This example illustrates one of the main limitations of MCQs, which is that the data collected using a regular MCQ does not distinguish between a student who knew the answer and one who guessed it, which is undesirable in a good assessment system.

Student models are usually used to infer students' knowledge states from learning data. For example, models, such as BKT [55] or Item Response Theory (IRT) [237], use observations of correct and incorrect answers to assess students' mastery of skills. BKT handles the possibility of students guessing the right answer by using a probability of guessing as a parameter of the

Level of confidence	Score if correct	Score if incorrect
High	3	-3
Medium	2	-1
Low	1	0

Table 8.1 – Example of scores attributed for multiple-choice answers depending of confidence levels

model. These different methods focus on improving statistical or Bayesian inference methods to deduce students' knowledge states or predict future performance. However, unless the problem of improving the data collection mechanism is addressed, the problem illustrated in the example mentioned above cannot be properly solved by better inference.

To counteract student guessing, a wide range of alternative scoring mechanisms have been proposed, such as normalising scores to remove the expected number of points earned by chance [33], giving negative points to wrong answers, allowing students to select multiple answers and penalising wrong selections [34], and allowing students to select multiple true-false items and scoring based on the number of correct assignments [129, 119]. It can be argued that most of these alternative marking schemes are improvements upon the common marking scheme [33].

Partial Knowledge

When facing a MCQ, it is not uncommon that students are able to eliminate some of the options but not all. Alternative scoring mechanisms that allow one to measure partial knowledge lead to less guessing from students because students, who would otherwise guess, are now able to receive credit for their partial knowledge [181, 241]. Another approach to measure partial knowledge has been to use IRT to estimate differences between incorrect answers [226]. Additionally, partial credit has been shown to bring potential improvements for Knowledge Tracing algorithms by going beyond binary prediction of students' knowledge or performance [169].

8.1.2 Estimating Confidence

Another particular type of alternative scoring mechanism for MCQs is to use students' reported level of confidence [33, 59, 119, 236]. The idea behind confidence levels is that when answering with high confidence, students will receive higher positive points if they are correct but also a higher penalty if they are incorrect. On the other hand, low confidence answers lead to low potential gain when correct and low penalty when incorrect, thus yielding less risk.

Table 8.1 shows an example of scoring that can be used for MCQs with confidence levels. Interestingly in this example, students will never leave the question blank because the lowest confidence level can only give positive points. The common practice is to let the student select both an answer and the level of confidence directly. However other methods have been also successful, such as asking the level of confidence after showing the question, but before showing the multiple choices [59]. Collecting confidence levels reduces guessing because students who would guess instead decide to select the low risk option and answer with low confidence. In practice, we would like students to answer with high confidence when they are truly confident and with low confidence when they are not. This is a particular property of Proper Scoring Rules that we discuss in the next section.

8.1.3 Changing Students' Input

As we focus on mechanisms to collect data from students, it is important to point out that some of these scoring schemes influence the data that is collected. We identify two main sources of such changes. The first is the scoring rule. How points are attributed changes how students decide to answer the questions. An example of scoring influencing students' answering strategies is the use of negative marks ($+1$ for a correct answer and -1 for an incorrect answer). This has been shown to reduce guessing but also leads to students deciding to leave questions blank [34]. If we assume that students answer in such a way as to try to maximise their expected score to the best of their knowledge, their answering strategy would be to select an answer if and only if they consider it more likely to be the correct answer than not.

The second source of change is the input of the answering mechanism. For example, the system could allow students to select more than one choice or to select both an answer and a confidence level. Thus, the data collected is not only one choice per student per question. In the following section we focus on probabilistic tests where students answer MCQs by selecting a probability for every choice. Changing the students' input can be beneficial if the new collected data proves to be useful. This kind of change is of particular interest for any data analysis that will be done. For analysing such data and extracting students' knowledge states, it is important to understand how the answering and scoring mechanism influence students.

8.2 Probabilistic Multiple-Choice Questions

In this section, we describe and analyse the concept of probabilistic testing [181, 195]. Probabilistic tests are very similar to MCQs. The difference is that instead of selecting one choice that the student thinks is the most likely to be true, the goal of a probabilistic test is that students report how likely they think each answer is to be the correct answer. This is done by letting the student assign a probability to every possible answer (see Figure 8.2 for an example of probabilistic test interface). First, we describe the multiple advantages of probabilistic testing. Then, we focus on the concept of proper scoring rule, which is central to probabilistic tests. Finally, we share preliminary observations obtained from an online web application for probabilistic quizzes.

8.2.1 Advantages of Probabilistic Tests

Because of their similarity with MCQs, probabilistic tests share the same advantages of efficiency and scalability and require the same preparation. Interestingly, probabilistic tests are more time consuming to grade by hand but can also be automatically graded if using computerised testing [181].

Compared to a MCQ with the same number of questions, probabilistic testing increases assessment reliability although students need more time to answer the questions [195]. However, these results may strongly depend on the students' mastery of probabilistic tests. Probabilistic testing may be unreliable for younger students with low or no understanding of probabilities, but more advantageous if the students can be taught the mechanics of the probabilistic answers beforehand [181].

Probabilistic testing allows one to estimate students' confidence. Students highly confident will answer with a probability close to 100% and the whole range of probabilities can measure all levels of confidence of the students. It should be noted that answering 0% means that the student is very confident that this choice is not the correct answer. For a question with 4 choices, the probability assignment for the lowest level of confidence would be 25%.

When answering a MCQ, the students may not know the answer, but are able to know that one answer is more likely than the others or to eliminate one or several wrong answers. Probabilistic MCQs allow one to express such partial knowledge. Some example of students' knowledge states could be:

- The student has sufficient knowledge and assigns a large probability to the correct answer.
- The student has not fully acquired the learning goals and answers 70% to the correct answer and 10% to every other choice.
- The student has partial knowledge and is able to eliminate two choices but hesitates between the two remaining choices.
- The student has incorrect knowledge and assigns a large probability to a wrong choice.
- The student has no knowledge and assigns equal probability to every choice.

8.2.2 Proper Scoring Rules

Proper Scoring Rules are scoring mechanisms for probabilistic judgments that attribute partial scores to partial knowledge in such a way that students are encouraged to honestly report their partial knowledge and not to guess. Proper scoring rules are mostly used for evaluating forecasting in prediction markets [37, 72]. They have also been used for assessments within the context of education to score probabilistic tests [20, 125, 216].

How the students' answers are scored is a critical part of an assessment mechanism. In particular, it influences what answers students input in the learning environment [216]. For probabilistic testing, the students must input a probability for every available choice. For a MCQ with m possible answers, the student submits a vector $a_1,\dots,a_m$ of probabilities. In general, the system would also enforce that $\sum_i a_i = 1$. A scoring rule would simply be a function S that takes the index correct answer i and the vector of probabilistic answers $\mathbf{a}$ and computes a score $S(i,\mathbf{a})$. Consider the simple scoring rule $S(i,\mathbf{a}) = a_i$. If a student assigned 70% probability to the correct answer, the student would receive 70% of the mark. Now, if a student hesitates between two choices A and B and thinks that the correct answer has 70% chance to be A and 30% to be B. If the student seeks to maximise the expected score, then the student will choose to answer (a_1, a_2) that maximises the $0.7S(1,\mathbf{a}) + 0.3S(2,\mathbf{a}) = 0.7a_1 + 0.3a_2$. The answer that maximises the student's expected score is $(a_1, a_2) = (1,0)$. It is unfortunate that in the scenario described above, the student who rationally tries to maximise the expected score does not answer honestly with a representation of his or her partial knowledge. We would like the scoring rule to reward students for answering with their exact partial knowledge. Proper scoring rules have such a property [37, 72].

Unlike the example mentioned above, some scoring rules have the property given on Equation 8.1 where $\mathbf{p}$ is a probability distribution over the possible answers to the question. This property enforces that a student who believes that the choices are likely to be correct with probability $\mathbf{p}$ and seeks to maximise the expected score will submit the answer $\mathbf{a} = \mathbf{p}$. Scoring rules with this property have been called Reproducing Scoring System [216] or proper scoring rules [20, 72]. The term Reproducing Scoring System was chosen because the aim of such a scoring system is that students will reproduce in their answer their exact state of knowledge, assuming they have the ability to do it accurately.

$$\max_{\mathbf{a}} \sum_i p_i S(i,\mathbf{a}) = \sum_i p_i S(i,\mathbf{p}) \tag{8.1}$$

Proper Scoring Rules have sparked some interest and have been used for probabilistic testing [83, 181] and in a course on decision making under uncertainty [20]. Another particularity of Proper Scoring Rules is also their sensitivity to students' knowledge compared to regular scoring mechanisms (e.g., giving full-mark for a correct answer and zero for an incorrect answer) [20]. For example, imagine three students (Alice, Bob and Charlie) answering a True-False question. They respectively think that the question statement is true with probability 100%, 51% and 49%. The regular scoring mechanism will lead them to answer respectively true, true, and false. Alice and Bob give the same answer, so, according to the regular scoring mechanism, they will receive the same score despite having very different knowledge states. On the other hand, Bob and Charlie have very similar knowledge states because they are both very close to completely guessing. The regular scoring mechanism in that situation does not correctly reflect the similar knowledge states of Bob and Charlie. This problem reveals a weakness of MCQs, which can fail to collect satisfactory information to efficiently assess students' knowledge states. However, a mechanism relying on a proper scoring rule and probabilistic answers would give Bob and Charlie a similar score very different from the score that Alice would receive.

Quadratic Scoring Rule

Several scoring rules have the property defined in Equation 8.1. The most used are the Logarithmic, Spherical and Quadratic scoring rules [195, 216]. Although the choice of proper scoring rule has sparked interest among researchers [20, 196], we will not discuss these nuances. We believe that the advantages of probabilistic testing have far more impact than the nuances in the choice of a proper scoring rule. Below, we provide the definition of the Quadratic Scoring rule because it is implemented in the quiz answering application that we describe in the following section. Additionally, we prove that it has the property of proper scoring rules.

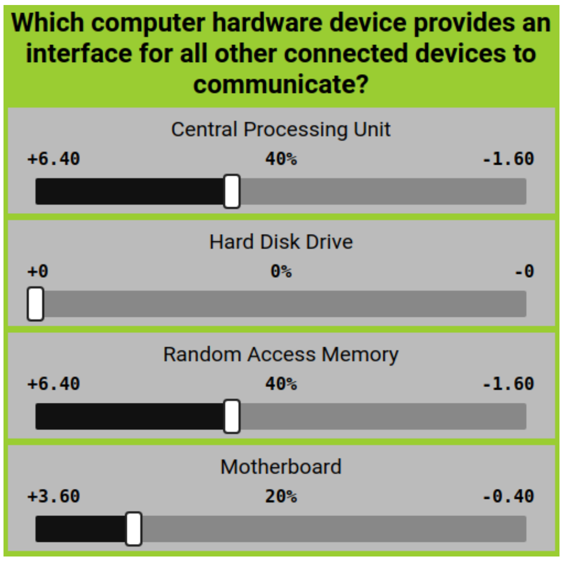
The quadratic scoring rule is defined by the formula given in Equation 8.2. We can observe that the probability assigned to the correct answer contributes $(2a_i - a_i^2)$ to the score of the student, while every probability assigned to wrong choices contributes a negative score of $-a_j^2$.

$$S(i, \mathbf{a}) = 2a_i - \sum_j a_j^2 \tag{8.2}$$

If the student believes with probability p_i that the choice i is the correct answer, the student will expect to receive the score $p_i(2a_i - a_i^2) + (1 - p_i)(-a_i^2)$. This can be simplified into $2a_i p_i - a_i^2$ for which one can easily verify that the maximum expected score is obtained for $a_i = p_i$. This shows that the Quadratic rule is indeed a proper scoring rule. Interestingly, this rule also allows us to give a score to answers for which the sum of the probabilities is not equal to 1. However such answers would always be sub-optimal and students will maximise their expected score if they answer honestly with their exact level of confidence.

8.2.3 Preliminary Observations

As a first step to analyse the potential of probabilistic testing, we developed an online application for answering probabilistic quizzes. Using this application, users can either create quizzes and share them online using a randomly generated URL or simply open such a quiz and answer the questions in sequence. The interface to answer questions is shown on Figure 8.2. The interface consists of a question at the top and a list of choices in randomised order below, as is usual in MCQs. For each choice, a slider allows a participant to select any probability between 0% and 100% by steps of 5%. Users must submit probabilities that sum to 100%. For


Which computer hardware device provides an interface for all other connected devices to communicate?

Central Processing Unit
+6.40 40% -1.60

Hard Disk Drive
+0 0% -0

Random Access Memory
+6.40 40% -1.60

Motherboard
+3.60 20% -0.40

Figure 8.2 – Screen capture from the probabilistic question answering interface

each choice, the number of points that will be earned or lost are shown on the sides.

The online application contains a large number of quizzes on a wide range of topics including mathematics, physics, general knowledge and computer science. Because the application was shared on social networks, such as Twitter and YouTube, several of the quizzes received answers from more than 100 participants, which allowed us to run preliminary analysis that we report in this section.

We focus on two different aspects. Firstly, we observe that the probabilistic tests have the potential to solve the issue described by Figure 8.1. Secondly, we observe that the probabilistic answering mechanism allows the learning environment to extract meaningful partial knowledge states of students.

Comparison of Probabilistic and Traditional MCQ

We mentioned previously that traditional MCQs can lead to situations where the teacher would not be able to distinguish between different knowledge states of the students in a classroom because these different states would generate the same observations. This concept is also similar to the concept of identifiability that occurs when different values of the parameters of a student model generates the same predictions of performance [19]. Probabilistic tests have the potential to produce additional information about the students' knowledge states. Thus, different states of the student knowledge would not generate similar observations.

This can be observed in the example of the question given on Figure 8.3. The figure shows the distribution of students' answers on two questions (blue and orange). On the left is the distribution of answers that would be observed using a traditional MCQ answering mechanism.

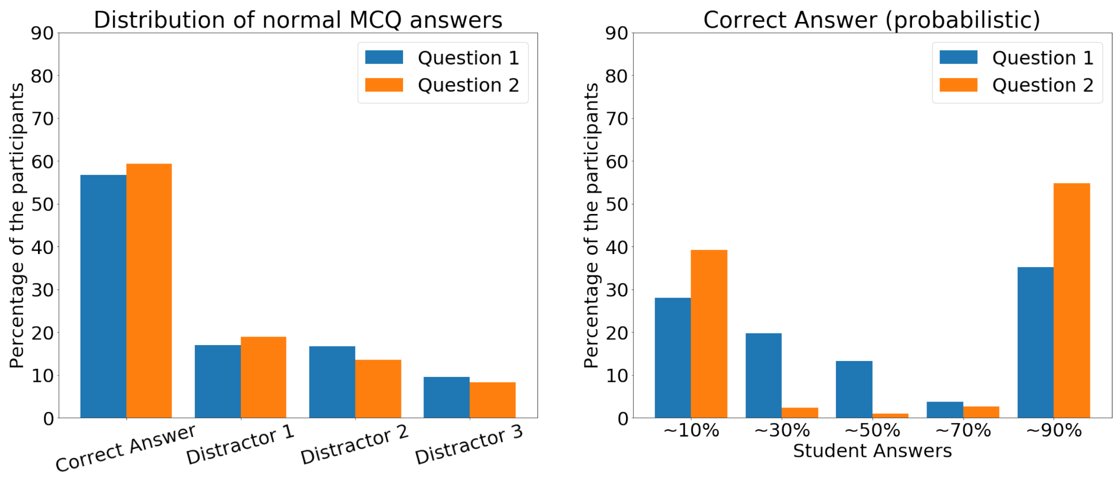


Figure 8.3 – Comparison of the measure between two questions. The question gives very similar results in terms of traditional MCQ statistics and very different results in terms of students probabilistic answers.

For this, we simulated the answers that participants would have given by assuming that students seek to maximise their expected score. Thus, in a traditional MCQ test, students would have answered the choice to which they gave the highest probability of being the correct answer. If two choices received equally high probabilities their answer is split randomly. On the right is the distribution of probabilities assigned to the correct answer.

The two questions have a very similar distribution of answers for the normal MCQ but not for the probabilistic answers. The number of answers with 30% and 50% probability from students on the correct choice are very different between the two questions. The first question (blue) had many students answering with low confidence while the second question (orange) had most students answering with probabilities concentrated on either 0% or 100%. As a consequence, teachers using normal MCQs would not be able to observe any difference between the two questions (similarly as the problem illustrated by Figure 8.1), but teachers using probabilistic MCQs would. Overall, we found 157 out of 903 pairs of questions for which the differences observed with the probabilistic MCQ concerned at least 10% more students as the differences observed using traditional MCQs.

Extracting Students' Partial Knowledge States

As students' answers to probabilistic MCQs are a vector of probabilities, Researchers can use mathematical tools to analyse families of vectors, such as clustering or dimension reduction. In the case of probabilistic testing, extracting knowledge states is similar to performing a clustering of the students' answers into groups with similar knowledge states. This can be performed using K-means algorithm [140]. Figure 8.4 shows an example of clusters extracted by the K-means algorithm on one question with three distractors. When observing the an-

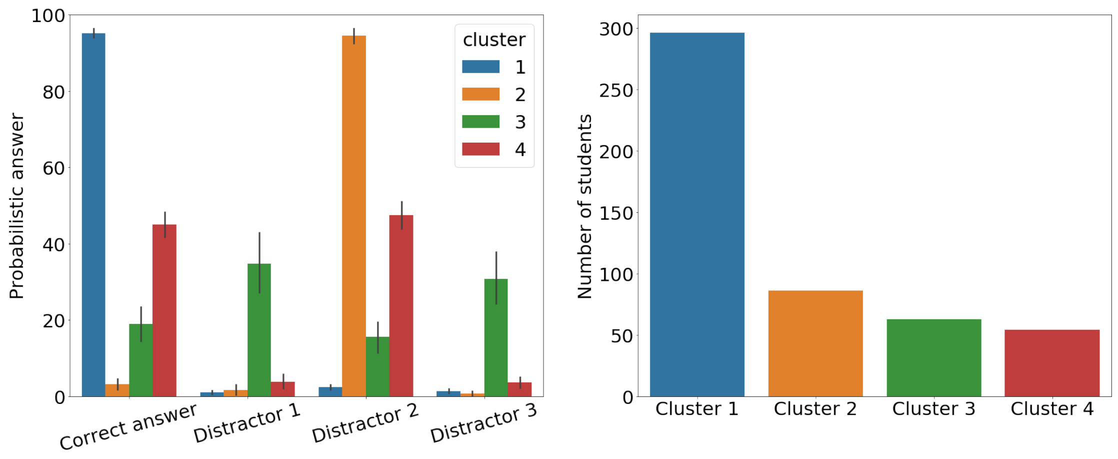


Figure 8.4 – Example of cluster extraction for a question with 4 choices; (top): average probability answered for each choice by participants of each cluster; (bottom): distribution of the participants within the four clusters

swers of students in each of the three clusters, we find specific clusters representing different knowledge states of the students.

Cluster 1 (blue) includes the students who assigned a very high probability to the correct answer. Cluster 2 (orange) includes the students who were very confident that the second distractor was the correct answer. Cluster 3 (green) includes mostly the students that did not know the answer at all and answered approximately 25% to each of the four choices. Cluster 4 (red) includes the student who hesitated between the correct answer and the second distractor and answered approximately 50% for the two choices. Clusters 3 and 4 represent knowledge states of students that would not be perceived using the traditional MCQ answering mechanism. For example, if the students from cluster 4 were given a regular MCQ, they would either answer correctly by luck when they were unsure or answer wrongly. In both cases, if a feedback mechanism relies on traditional MCQs, they would likely not receive feedback adapted to their exact knowledge.

8.3 Improving Learning Environments for Inductive Teaching

In this section we bring our focus back to inductive teaching and, in particular, the reasoning model that we defined in Chapter 7. In this model, students reason probabilistically about the categorisation tasks. We have shown that a particular mechanism in an adaptive learning environment for inductive teaching is to estimate the students' inductive biases in order to optimise the selection of teaching examples. In this section, we explore how the adaptive teaching strategy can benefit from probabilistic testing.

If we assume that students have the ability to perfectly relate their knowledge through the probability estimates of a probabilistic test, we can conclude that the observation of students in the adaptive strategy presented in Chapter 7 could be replaced by a single step that would be a direct estimation of students' inductive bias. This step would be followed by the optimal teaching strategy when the learning environment has no uncertainty about the student. Unfortunately, students in general do not master the mechanism of probabilistic testing sufficiently for this assumption to be true [181]. Probabilistic tests are difficult if students have never experienced answering such tests, especially for younger students [195]. However, students are able to improve in answering such tests [83].

Even though the scoring rule in Equation 8.2 gives students the incentive to answer honestly and to the best of their knowledge (Property 8.1), it was shown that students, due to a lack of experience with the probabilistic answering mechanism, diverge from the most rational answering behaviour. In this section, we analyse the consequences of different assumptions describing how students answer probabilistic tests.

Noisy Dirichlet Answers

Our first approach to analyse the effect of probabilistic testing for inductive teaching is to assume that students' answers would randomly deviate from their exact state of knowledge $\mathbf{p}$. For this, we model the observations as a random variable. Because the students' answers $\mathbf{a}$ are distributions of probabilities, it is natural to model the randomness as a Dirichlet distribution. Additionally, the parameters of the Dirichlet distribution should be chosen in order to represent that the noisy observations are deviations from the student's knowledge state $\mathbf{p}$. Such parameters must be proportional to $\mathbf{p}$. We define $\mathbf{a} \sim Dirichlet(M\,\mathbf{p})$, where M is a positive real number.

Larger values of the parameter M will consist of very small deviations while the smaller values will generate more noisy answers. We discuss at the end of this subsection how different values for M influence students' noisy answers and which values are best suited to correctly model students' probabilistic answers.

In order to simplify the notations, we will not use the notations B_{i^*} and B_i as in Chapter 7 for the unknown inductive bias of the student and an incorrect other inductive bias but $\mathbf{p}$ and $\mathbf{q}$ instead. The result obtained in the previous chapter can be re-written as Equation 8.3.

$$\mathbb{E}[\log(\frac{P(S_\mathbf{p} \mid c)}{P(S_\mathbf{q} \mid c)})] = \log(\frac{P(S_\mathbf{p})}{P(S_\mathbf{q})}) + D_{KL}(\mathbf{p} \| \mathbf{q}) \tag{8.3}$$

In the case of probabilistic testing, the observed student answer is not a category c anymore, but the observed probability vector $\mathbf{a} \sim Dirichlet(M\,\mathbf{p})$. In this section, we compare the posterior expected information about the students' inductive bias $\mathbb{E}[\log(\frac{P(S_\mathbf{p} \mid \mathbf{a})}{P(S_\mathbf{q} \mid \mathbf{a})})]$ after observing a probabilistic answer with the one obtained after observing the choice of a single category given in Equation 8.3.

The probability density of the noisy observations $\mathbf{a}$ can be computed using the Gamma function $\Gamma(\cdot)$ as given by Equation 8.4.

$$p(\mathbf{a}|S_{\mathbf{p}}) = \frac{\Gamma(M)}{\prod_i \Gamma(M\,p_i)} \prod_i a_i^{(Mp_i-1)} \tag{8.4}$$

Using the expression in Equation 8.4 and applying Bayes rule for conditional probabilities, we can derive the expectation given in Equation 8.5.

$$\mathbb{E}[\log(\frac{P(S_{\mathbf{p}}|\mathbf{a})}{P(S_{\mathbf{q}}|\mathbf{a})})] = \log(\frac{P(S_{\mathbf{p}})}{P(S_{\mathbf{q}})}) + \log(\frac{\prod_i \Gamma(M\,q_i)}{\prod_i \Gamma(M\,p_i)}) + M\sum_i (p_i - q_i)\mathbb{E}[\log(a_i)] \tag{8.5}$$

In the case of sufficiently large values of M, the expression above can be simplified in two ways. Using the approximation $\log(\Gamma(z)) \simeq z\log(z) - z$, the term $\log(\frac{\prod_i \Gamma(M\,q_i)}{\prod_i \Gamma(M\,p_i)})$ can be simplified to $M\sum_i [q_i \log(q_i) - p_i \log(p_i)]$. Additionally, using the approximation of the Digamma function $\psi(z) \simeq \log(z)$, we obtain the second term $\mathbb{E}[\log(a_i)] = \psi(M\,p_i) - \psi(M) \simeq \log(p_i)$. Finally, we obtain the simplified result in Equation 8.6.

$$\mathbb{E}[\log(\frac{P(S_{\mathbf{p}} \mid \mathbf{a})}{P(S_{\mathbf{q}} \mid \mathbf{a})})] \simeq \log(\frac{P(S_{\mathbf{p}})}{P(S_{\mathbf{q}})}) + M\,D_{KL}(\mathbf{q}\|\mathbf{p}) \tag{8.6}$$

The approximations we have used are not valid for small values M. This is not important because, as we explain later, small values of M correspond to students who do not have the ability to answer probabilistic tests. In the case of large M, Equation 8.6 gives an estimation of the uncertainty in a similar form as in Equation 8.3. However, the KL divergence in Equation 8.6 is multiplied by the parameter M which means that, if this parameter is large, the estimation of the student's inductive bias will be faster by a factor of M.

Figure 8.5 shows the distribution of answers expected from the noisy answering behaviour that we defined above. The four histograms give the distribution of the answer of a student with partial knowledge $\mathbf{p} = (0.75, 0.2, 0.05)$ obtained with four different values of the noise parameter M. We aim to give an idea of what values for the parameter M models students best given their level of mastery of the probabilistic answering mechanism. On the figure, we can observe that for the parameter value $M = 0.1$, the answering behaviour consists of nearly always assigning 100% to one category that is chosen randomly following the probability distribution $\mathbf{p}$. This behaviour is identical to the expected answers that we analysed in Chapter 7. For $M = 1$, the student very often answers 0% for both the choices to which the partial knowledge assigns 20% and 5% probability. For $M = 10$, we observe that the student still

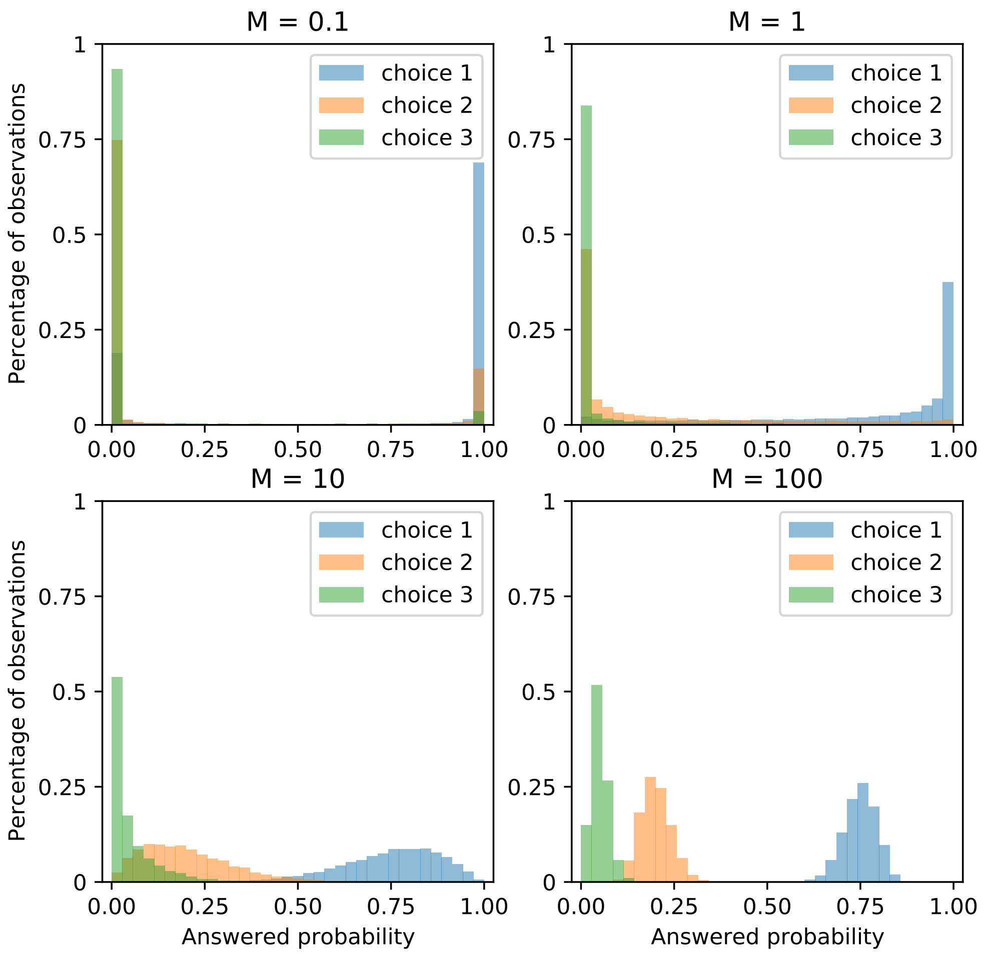


Figure 8.5 – Examples of student answering behaviour depending on different values of the parameter M

provides very noisy answers, but is closer to the partial knowledge than for the smaller values of M. The spread of answers for the 75% confidence ranges from 50% to 100% and the answers for the 5% confidence choice are very often 0%. For $M = 100$, we find that the student is much better at reporting the knowledge state **p** as the distribution of answers are well centered around the correct values of the student's partial knowledge.

It seems realistic to assume that students will provide an answering behaviour similar to the noise level for $M = 10$. In that case, the learning environment can expect to estimate the students' inductive biases ten times faster than using traditional questions. However, as previous studies on probabilistic testing indicated, the answers given to probabilistic tests strongly depend on the students' ability. Behaviour such as the one described by the parameters $M = 0.1$ and $M = 1$ cannot be neglected. Finally, answers of the type described by the parameter $M = 100$ will only be obtained from students who have experience with the answering mechanism and a good understanding of probabilities. In this case, the probabilistic test is as informative for the adaptive learning environment as querying the student 100 times.

Overconfidence and Underconfidence

Previous research has revealed specific types of deviation from the perfect reporting of partial knowledge in probabilistic testing. Depending on the context, probabilistic answers have been shown to be underconfident (for example in risk auditing [234]) or overconfident (for example, in learning vocabulary [125] or in medical diagnosis [100]). In this subsection, we analyse how the students' inductive biases can be inferred from either overconfident or underconfident answers.

For binary questions, overconfidence pushes the answered probability away from 50% towards 100% or 0%. On the other hand, underconfidence pulls the students' answers towards 50%. Overconfident and underconfident answers can be modelled probabilistically. This can be done by using the identity $p = (1 + e_p^{\theta})^{-1}$. For values of θ with large absolute value, the probability is close to 0% or 100%, and for values of θ with low absolute values, the probability p is close to 50%. In other words, overconfidence will consist of answering with higher absolute values for the parameter θ than the students' exact partial knowledge state while underconfidence will be the opposite.

We can assume random noise from a normal distribution $\epsilon \sim \mathcal{N}(\mu, \sigma)$ and the observed answers are $a_p = (1 + e^{\theta_p + \epsilon})^{-1}$. For the learning environment to correctly estimate the students' inductive biases, we must compare two distributions a_p and a_q obtained in this way from two different possibilities for the student's partial knowledge. We obtain the KL-divergence given in Equation 8.7.

$$D_{KL}(a_p \| a_q) = \frac{1}{\sigma^2}[\log(\frac{1-p}{1-q}) - \log(\frac{p}{q})]^2 \tag{8.7}$$

We find that the KL-divergence is larger than in the case of querying for a choice of category, except for large values of the noise variance σ^2. Thus, the probabilistic test would, in this case, allow for faster estimations of the students' inductive biases. However, if the overconfidence or underconfidence deviations have an impact too large on students' answers, it is preferable to ask the student to pick a category rather than to provide a probabilistic answer.

Discussion

We have shown in this section that probabilistic testing can bring large benefits for the estimation of students' inductive biases. These benefits, however, depend on the students ability to correctly use the probabilistic answering mechanism. If students submit very noisy answers, or if students are very overconfident, it is preferable not to use probabilistic tests. These aspects could be included within the adaptive learning environment.

The adaptive strategy described in Section 7.3.2 in the previous chapter could be modified to allow for the possibility of using probabilistic tests. Because probabilistic questions take a longer time to answer [195], the optimal adaptive policy could decide to either use normal or probabilistic questions depending on the amount of information that is expected to be gained from each mechanism and the time available. Additionally, the decision to use probabilistic testing for a given student could depend on the learning environment's expectation of the student's ability to answer probabilistic tests.

8.4 Conclusion

In this chapter, we described an answering mechanism and scoring rule for MCQs based on collecting probabilistic answers from students. Previous work showed the importance of alternative scoring mechanisms to alleviate the issue of students guessing the correct answers in a MCQ [33, 34]. We have shown that the advantages of alternative mechanisms for MCQs can go beyond reducing guessing. Probabilistic tests yield better prospects for the assessment of partial knowledge and for estimating students' confidence. Additionally, probabilistic testing keeps most of the advantages of usual MCQs such as automated grading, simplicity of design and scalability.

Probabilistic testing could have several other advantages that would be interesting to evaluate. First, students will possibly develop the habit of correctly estimating their confidence or at least reducing their overconfidence, which we expect will help them develop metacognitive skills. Additionally, probabilistic answers could be used to provide automated feedback. Using normal MCQs, the learning environment (or teacher) can only adapt the feedback based on the selected choice, which, as we have shown, carries less meaningful information about students' knowledge than probabilistic answers. An adaptive feedback system based on regular MCQ answers will necessarily give the same feedback to a student who knows the answer and one who guessed it because they provide the same input to the system. Using the clusters such as the one we showed on Figure 8.4, feedback could be provided to each cluster of students in a fully automated way. Furthermore, it may be interesting to explore how different types of feedback should be given to students who are very confident in a wrong answer, students who hesitate between several choices and students who do not know the answer at all.

Although we reported only preliminary observations using the probabilistic answering mechanism, we also contributed theoretical results. We have shown in Section 8.3, under different assumptions about students' answering ability, that probabilistic testing is beneficial to the adaptive teaching algorithms that we proposed in Chapter 7. The supplementary information obtained from the probabilistic answers allows us to more efficiently estimate students' inductive biases.

9 Conclusion

This thesis started from the observation that data-driven digital technology is transforming education in many areas. A very positive impact of this transformation is the numerous opportunities to improve education through bodies of research emerging from this change. These bodies of research include Learning Analytics, Educational Data Mining, Learning at Scale, Technology-Enhanced Learning, and Computer-Supported Collaborative Learning. All these fields have indisputably brought new practices and technology that have benefited teachers and students in multiple ways. Yet, the current state of educational technology is nowhere near an optimal point and improvements can still be made in many domains.

For this thesis, we chose to use the opportunities given by data-driven digital technology to contribute to adaptive and self-improving learning environments. Moreover, our efforts were guided towards supporting inductive teaching methods. Specifically, we focused on how a learning environment for inductive teaching can model students' states and provide personalised instruction while improving itself over time. Inductive teaching has been proven to be an educational practice very beneficial to students, however, within adaptive teaching, less research has focused on investigating and supporting inductive methods than deductive methods.

9.1 Contributions

Our contributions in the first part of the thesis relate to student modeling, adaptive teaching and self-improvement of learning environments. In Chapter 2, we contributed to student modeling and adaptive teaching. We reported our study on predicting progress rate and time completion in several classrooms. This study evaluated methods for modeling the state of students and anticipating how this state changes over time, which is a required aspect in any adaptive learning environment. The system we implemented for this study combined multiple aspects of learning analytics, including the data collection, data analysis, and use of the data in real time through a teacher dashboard. In Chapter 3, we focused on student models and, in particular, generative models that allow us to simulate students' performance or behaviour. We reported our study on the simulation of MOOC students' behaviour. Our main contributions are the design and the evaluation of a student model using Semi-Markov chains. We showed that the Expectation-Maximisation algorithm can be efficiently used to estimate latent parameters of probability distributions for such generative student models. In Chapter 4, we focused on the concept of self-improvement. Our contribution is the adaptation of Multi-Armed Bandit (MAB) optimisation to arbitrary Bayesian student models in order to improve the process of selection of pedagogical content or learning activities in learning environments over time.

In the second part of the thesis, we brought our focus on inductive teaching. In Chapter 5, we reported our study on inductive reasoning. In this study, we assigned students to solve categorisation questions where unlabelled examples had to be assigned to different categories of labelled examples. Our study revealed that students exhibit a bias towards classifying the examples according to some particular feature. Most importantly, we found that students exhibit individual differences in their inductive biases. This finding is central to our contributions, because it shows that a learning environment using inductive teaching must adapt to the individual biases of the students to teach optimally. Additionally, we studied how receiving feedback changed the students' inductive biases and found that students also differed individually on this aspect. In Chapter 6, our contribution is a Bayesian student model that predicts students' answers during categorisation tasks, traces over time the inductive biases of students, and accounts for changes in their biases. The model achieved a large improvement over a baseline prediction. In Chapter 7, our contributions are computational methods for optimal inductive teaching, optimal inductive teaching with uncertainty about the students' biases, and self-improving teaching strategies. The contributions from Chapter 6 and Chapter 7 are the first adaptive and self-improving computational methods targeted at personalising inductive learning activities. These contributions can be integrated within an Intelligent Tutoring System to support inductive teaching practices.

Finally, in Chapter 8, we focused on probabilistic testing. It is a particular data collection mechanism for Multiple-Choice Questions. Several advantages of probabilistic testing include reducing students' guessing, estimating students' confidence and rewarding partial knowledge. Probabilistic testing is of interest for the thesis for two main reasons. First, improving the collection of student data is complementary to improving inference mechanisms in adaptive and self-improving learning environments. Second, as we modeled the students states in inductive learning activities using probability distributions, directly asking for probability estimates from students is a particularly good fit to our modeling approach. Our contribution was to show that, assuming some level of mastery of the probabilistic answering mechanism, probabilistic testing would indeed allow a learning environment to more efficiently adapt to students' individual differences in inductive reasoning.

9.2 Limitations

This thesis set the goal of developing adaptive and self-improving learning environments for inductive teaching. It is fair to say that our contributions have been mostly theoretical. Although we implemented inductive learning activities, these aimed only at evaluating aspects of inductive reasoning and testing our student model. The complete development of such a learning environment is a challenging endeavour for which we only contributed algorithmic foundations for student models, adaptivity and self-improvement.

Our main contributions correspond to the steps 1, 4, and 5 of the framework that we describe in the introduction (see Figure 1.1) and are not sufficient for the development of a functional learning environment. Instead, our contribution should be understood as an external optimisation engine. One of the most important missing steps is to provide learning activities. Providing learning activities would require two additional efforts. First, we must choose the concepts to be taught inductively and the examples to support the induction process. Second, we must design efficient user interfaces for inductive reasoning. The thesis contained examples of concepts (Chapter 5) and an example of user interface (see Figure 7.4), but these were designed only for the purpose of experiments. More research about these two aspects is needed to complete the thesis contributions.

We have described some possibilities for self-improvement using MAB optimisation. This contribution is limited in two ways. First, other approaches in reinforcement learning could be considered. MAB optimisation is mostly useful in the case of a task that is repeated over a number of independent and identical steps. This will cover the majority of cases for a learning environment that interacts with a large number of students over a long period of time (for example, a MOOC platform or an online learning web application). Yet, more general approaches from the field of Reinforcement Learning could be used to increase the number of possibilities for self-improvement. Second, we only considered some ways in which the learning environment can self-improve. In Chapter 4, our contribution allows us to apply MAB optimisation to the particular case of Bayesian student models for estimating transition probabilities of different learning activities. In Chapter 7, we have shown that similar optimisation can be used to increase learning outcomes of students while a learning environment still needs to collect data about students in order to optimise an adaptive teaching strategy. Arguably, many more aspects of the learning environment could be subject to self-improvement using similar methods as the one we have proposed.

Another limitation of our contributions is that the optimal teaching strategies that we defined in Chapter 7 will in practice only be as good as the student model they are optimising for. Although it is supported by previous research and partially by the accuracy gains of our Bayesian student model in Chapter 6, the inductive reasoning model is certainly not a perfect representation of students' reasoning. Instead, we expect that the model will be refined and improved as more data becomes available. Similarly, improvements of BKT have been developed over the years.

Throughout the thesis, we focused mostly on short-term changes in the students' inductive biases, but long-term effects are indisputably worthwhile to investigate too. We studied changes, either due to negative feedback on their choice of classification (Chapter 5) or by showing examples and counterexamples during a learning activity (Chapter 7). Additionally, we have shown that changes due to feedback decay over a short period of time. Both how students' inductive biases and their reasoning ability evolve over long periods of time would be interesting research areas that would complete the contributions of this thesis.

Finally, our analysis of probabilistic testing in Chapter 8 included multiple limitations. First, our observations did not come from an educational context. Second, the observations did not result from an experimental condition. Third, our conclusions on the superiority of probabilistic testing for personalised inductive teaching rely on assumptions about students' mastery of the answering mechanism that we should evaluate. We included such preliminary work in the thesis for two reasons. First, it completed our contributions because the data collection mechanism corresponds to the step 3 of the framework that we describe in the introduction (see Figure 1.1). Second, it is exceptionally fitted to our student model and teaching strategies from Chapter 7.

9.3 Next Steps

We envision two main research projects to push the contributions of the thesis further. The first project is the development of an inductive learning environment that uses the adaptive and self-improving algorithms and would be deployed in classrooms. The second project is to pursue the analysis of probabilistic testing, notably in the context of inductive reasoning.

Implementing the contributions of the thesis in a learning environment that would be used by instructors and students would bring multiple benefits as a research project:

- **Evaluating inductive teaching over a wide range of different concepts.** In a learning environment that allows teachers to use the optimised teaching strategies for teaching multiple concepts to their students and assessing their learning outcomes, researchers can evaluate which concepts are best taught with inductive teaching approaches. Moreover, in this context, our proposed adaptive teaching strategies would be able to evaluate for which concepts students exhibit the most individual differences.

- **Developing interfaces to support students' inductive reasoning.** A second compelling research direction would be to develop and analyse support for students' inductive reasoning in the form of user interfaces. This project would answer several interesting questions such as: How should the learning environment present examples? How should the learning environment support students to think about multiple hypotheses? How should the learning environment support students to update their belief when observing examples?

- **Improving the contributions of the thesis.** As we have stated previously, our contributions rely on a model of students' inductive reasoning that necessarily makes simplifying assumptions about students' reasoning process. To evaluate the quality of our student model, we collected data in classrooms, but using experimental activities that can be considered very different than usual inductive activities. The implementation of an inductive learning environment will allow researchers to collect student data outside of an experimental setting. This data can then be used to improve the student model, the adaptive teaching strategy and the self-improving mechanism that we proposed in this thesis.

The second compelling next step is to study in depth the probabilistic assessment mechanism that we described in Chapter 8. We believe this project could be beneficial to research and to the design of adaptive learning environments for inductive teaching. We identify several interesting research directions:

- **How do students deviate from optimal rational answers?** It is not clear that students will correctly reproduce the exact state of their partial knowledge when answering probabilistic tests. We discussed in the thesis a few types of possible deviations, but these depend on the context and further understanding would be beneficial.

- **How can students improve at answering probabilistic tests?** Probabilistic tests have been shown to be difficult to answer for students without prior experience with probabilities and scoring mechanisms. Investigating how to help students best master this type of question answering would remove one of its main limitations.

- **Are students able to make calibrated probabilistic answers?** If a student answers 20 times with the probability 70%, it is expected that the selected choice is the right answer 14 times, and a wrong answer six times. If the student is right less than 14 times, we can infer that the answers were likely overconfident. The calibration of probabilistic answers measures whether the frequency of correctness matches closely with the probabilities answered. It is an important metric for measuring the quality of probabilistic answers.

- **How much do adaptive strategies relying on probabilistic answers improve students' learning outcomes?** Based on assumptions about students answering behaviour, we have shown that probabilistic answers would allow an inductive learning environment to more quickly estimate students' biases. It would be interesting to verify in which contexts the assumptions hold and if adaptive strategies that rely on probabilistic questions truly perform better.

- **In the context of inductive reasoning activities, how do students update their answers when observing new data?** The model studied in the thesis assumes that students perform a Bayesian update when observing new examples. Once again, this model is a simplification of the complex reasoning mechanisms of students. Using probabilistic tests, one could observe students' answers both before and after seeing a piece of evidence and analyse how the answers changed.

- **Does practicing answering with uncertainty have other positive side effects on students?** Unlike traditional MCQ which arguably give to students the habit of guessing when they don't know the answer to a question, probabilistic tests motivate honest answers. To answer probabilistic questions, students are invited to reflect on their own knowledge and confidence before selecting an answer. It is interesting to evaluate if this reflection leads students to develop better metacognitive skills and lower their overconfidence. Other side effects of probabilistic tests could be expected and would be very interesting to investigate.

9.4 Conclusion

In this thesis we have provided the theoretical foundations required to build adaptive and self-improving learning environments for inductive teaching. Through three main studies we have shown and evaluated different approaches and algorithms to model students' states and their evolution through time and interactions with a learning environment. We have found individual differences in students' inductive reasoning that show the importance of personalisation for inductive teaching learning environments. Finally, we showed how to optimise inductive teaching strategies and how probabilistic testing can benefit inductive teaching.

Although inductive teaching has been shown to be a very beneficial pedagogical approach, no student model allowed for tracing students' inductive reasoning as well as BKT traces students' skill mastery in ITS. This thesis filled this gap and advanced student modeling methods for inductive teaching. BKT improved adaptivity in many intelligent tutoring systems, which have now been used by millions of students. Similarly, we hope that our contributions will motivate further research in adaptive instruction using inductive methods and will lead to the development of adaptive and self-improving learning environments for inductive teaching.

Induction, defined in the most general manner, consists of generalising from observations. It describes a fundamental process of learning, which is not restricted to children learning about geometrical shapes or categories of animals. Induction is used in our everyday life to understand our environment and adapt our behaviour. Induction is used by scientists from all domains whether it is to infer which general theories of physics are correct or which medical treatments are more efficient [228]. Induction is the main goal of machine learning algorithms, which generalise statistical patterns and can predict new unseen data.

This thesis was motivated by the epistemological depth of the process of induction and the potential for students to benefit from such teaching practices. We built the multiple components of adaptive and self-improving learning environments for inductive teaching.

We believe that our proposed models and optimisation algorithms have the potential to generalise to a wide range of uses within several types of learning environments. In particular, the modeling of students' knowledge states with probability distributions describing their uncertainty has interests beyond simple inductive learning tasks. Tracing how students' uncertainty is influenced by new information is a core aspect of teaching that has the potential to improve educational technology in many domains.

Bibliography

[1] Deepak Agarwal, Bee-Chung Chen, and Pradheep Elango. Explore/exploit schemes for web content optimization. In *2009 Ninth IEEE International Conference on Data Mining*, pages 1–10. IEEE, 2009.

[2] Kolos Csaba Ágoston, Péter Biró, and Iain McBride. Integer programming methods for special college admissions problems. *Journal of Combinatorial Optimization*, 32(4):1371–1399, 2016.

[3] Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on learning theory*, pages 39–1, 2012.

[4] Vinca Aisa, Angelina Prima Kurniati, and AW Yanuar Firdaus. Evaluation of the online assessment test using process mining (case study: Intensive english center). In *2015 3rd International Conference on Information and Communication Technology (ICoICT)*, pages 472–477. IEEE, 2015.

[5] Kathryn Alesandrini and Linda Larson. Teachers bridge to constructivism. *The clearing house*, 75(3):118–121, 2002.

[6] Vincent Aleven, Elizabeth A McLaughlin, R Amos Glenn, and Kenneth R Koedinger. Instruction based on adaptive learning technologies. *Handbook of research on learning and instruction*, pages 522–560, 2016.

[7] John R Anderson, C Franklin Boyle, and Brian J Reiser. Intelligent tutoring systems. *Science*, 228(4698):456–462, 1985.

[8] Arif A Anwar and AS Bahaj. Student project allocation using integer programming. *IEEE Transactions on Education*, 46(3):359–367, 2003.

[9] Ivon Arroyo and Beverly Park Woolf. Inferring learning and attitudes from a bayesian network of log file data. In *AIED*, pages 33–40, 2005.

[10] Peter Auer. Using upper confidence bounds for online learning. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 270–279. IEEE, 2000.

[11] Jose Manuel Azevedo, Ema Patrícia Oliveira, and Patrícia Damas Beites. How do mathematics teachers in higher education look at e-assessment with multiple-choice questions. In *CSEDU (2)*, pages 137–145, 2017.

[12] Adrià Puigdomènech Badia, Bilal Piot, Steven Kapturowski, Pablo Sprechmann, Alex Vitvitskyi, Daniel Guo, and Charles Blundell. Agent57: Outperforming the atari human benchmark. *arXiv preprint arXiv:2003.13350*, 2020.

[13] Ryan Baker, Albert T Corbett, and Kenneth R Koedinger. Detecting student misuse of intelligent tutoring systems. In *7th International Conference on Intelligent Tutoring Systems* (ITS2004), 30 August–3 September 2004, Maceió, Alagoas, Brazil, pages 531–540. Springer, 2004.

[14] Ryan Shaun Baker and Paul Salvador Inventado. Educational data mining and learning analytics. In *Learning analytics*, pages 61–75. Springer, 2014.

[15] Ryan SJD Baker and Kalina Yacef. The state of educational data mining in 2009: A review and future visions. *JEDM| Journal of Educational Data Mining*, 1(1):3–17, 2009.

[16] Girish Balakrishnan and Derrick Coetzee. Predicting student retention in massive open online courses using hidden markov models. *Electrical Engineering and Computer Sciences University of California at Berkeley*, 2013.

[17] Joseph Beck. Difficulties in inferring student knowledge from observations (and why you should care). In *Educational Data Mining: Supplementary Proceedings of the 13th International Conference of Artificial Intelligence in Education*, pages 21–30, 2007.

[18] Joseph E Beck. Engagement tracing: using response times to model student disengagement. In *Proceedings of the 2005 conference on Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology*, pages 88–95, 2005.

[19] Joseph E Beck and Kai-min Chang. Identifiability: A fundamental problem of student modeling. In *International Conference on User Modeling*, pages 137–146. Springer, 2007.

[20] J Eric Bickel. Scoring rules and decision analysis education. *Decision Analysis*, 7(4):346–357, 2010.

[21] Gautam Biswas, Krittaya Leelawong, Daniel Schwartz, Nancy Vye, and The Teachable Agents Group at Vanderbilt. Learning by teaching: A new agent paradigm for educational software. *Applied Artificial Intelligence*, 19(3-4):363–392, 2005.

[22] Kristen Blair, Daniel L Schwartz, Gautam Biswas, and Krittaya Leelawong. Pedagogical agents for learning by teaching: Teachable agents. *Educational Technology*, pages 56–61, 2007.

[23] Paulo Blikstein. Multimodal learning analytics. In *Proceedings of the third international conference on learning analytics and knowledge*, pages 102–106, 2013.

[24] Benjamin S Bloom. The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational researcher*, 13(6):4–16, 1984.

[25] Alejandro Bogarín, Rebeca Cerezo, and Cristóbal Romero. A survey on educational process mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(1):e1230, 2018.

[26] Eric Bonabeau. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences*, 99(suppl 3):7280–7287, 2002.

[27] Mina Shirvani Boroujeni, Kshitij Sharma, Łukasz Kidziński, Lorenzo Lucignano, and Pierre Dillenbourg. How to quantify student's regularity? In *Proceedings of the 11th European Conference on Technology Enhanced Learning* (EC-TEL 2016), 13–16 September 2016, Lyon, France, pages 277–291. Springer, 2016.

[28] Daniel Pierre Bovet, Pierluigi Crescenzi, and D Bovet. *Introduction to the Theory of Complexity*. Prentice Hall London, 1994.

[29] Sebastien Boyer and Kalyan Veeramachaneni. Transfer learning for predictive models in massive open online courses. In *Proceedings of the 17th International Conference on Artificial Intelligence in Education* (AIED 2015), 22–26 June 2015, Madrid, Spain, pages 54–63. Springer, 2015.

[30] Christopher G Brinton, Mung Chiang, Sonal Jain, HK Lam, Zhenming Liu, and Felix Ming Fai Wong. Learning about social learning in moocs: From statistical analysis to generative model. *Learning Technologies, IEEE Transactions on*, 7(4):346–359, 2014.

[31] Björn Brodén, Mikael Hammar, Bengt J Nilsson, and Dimitris Paraschakis. Ensemble recommendations via thompson sampling: an experimental study within e-commerce. In *23rd International Conference on Intelligent User Interfaces*, pages 19–29, 2018.

[32] Apostolos N Burnetas and Michael N Katehakis. Optimal adaptive policies for markov decision processes. *Mathematics of Operations Research*, 22(1):222–255, 1997.

[33] Martin Bush. Alternative marking schemes for on-line multiple choice tests. In *7th Annual Conference on the Teaching of Computing, Belfast*, 1999.

[34] Martin Bush. A multiple choice test that rewards partial knowledge. *Journal of Further and Higher education*, 25(2):157–163, 2001.

[35] Andrew C Butler. Multiple-choice testing in education: are the best practices for assessment also good for learning? *Journal of Applied Research in Memory and Cognition*, 7(3):323–331, 2018.

[36] Stephanie Campbell and Christopher H Skinner. Combining explicit timing with an interdependent group contingency program to decrease transition times: An investigation of the timely transitions game. *Journal of Applied School Psychology*, 20(2):11–27, 2004.

[37] Arthur Carvalho. An overview of applications of proper scoring rules. *Decision Analysis*, 13(4):223–242, 2016.

[38] Hao Cen, Kenneth Koedinger, and Brian Junker. Learning factors analysis–a general method for cognitive model evaluation and improvement. In *International Conference on Intelligent Tutoring Systems*, pages 164–175. Springer, 2006.

[39] Kai-min Chang, Joseph Beck, Jack Mostow, and Albert Corbett. A bayes net toolkit for student modeling in intelligent tutoring systems. In *International Conference on Intelligent Tutoring Systems*, pages 104–113. Springer, 2006.

[40] Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. In *Advances in neural information processing systems*, pages 2249–2257, 2011.

[41] Sven Charleer, Jose Luis Santos, Joris Klerkx, and Erik Duval. Improving teacher awareness through activity, badge and content visualizations. In *Proceedings of the 13th International Conference on Web-Based Learning* (ICWL 2014), 14–17 August 2014, Tallinn, Estonia, pages 143–152. Springer, 2014.

[42] Ed H Chi, Peter Pirolli, Kim Chen, and James Pitkow. Using information scent to model user information needs and actions and the web. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 490–497. ACM, 2001.

[43] Incheol Choi, Richard E Nisbett, and Edward E Smith. Culture, category salience, and inductive reasoning. *Cognition*, 65(1):15–32, 1997.

[44] Brian Christian and Tom Griffiths. *Algorithms to live by: The computer science of human decisions*. Macmillan, 2016.

[45] Constantinos Christou and Eleni Papageorgiou. A framework of mathematics inductive reasoning. *Learning and Instruction*, 17(1):55–66, 2007.

[46] Douglas Clark and Marcia C Linn. Designing for knowledge integration: The impact of instructional time. *The Journal of the Learning Sciences*, 12(4):451–493, 2003.

[47] Benjamin Clement, Didier Roy, Pierre-Yves Oudeyer, and Manuel Lopes. Online optimization of teaching sequences with multi-armed bandits. 2014.

[48] Benjamin Clement, Didier Roy, Pierre-Yves Oudeyer, and Manuel Lopes. Multi-armed bandits for intelligent tutoring systems. *Journal of Educational Data Mining*, 7(2), 2015.

[49] Carleton Coffrin, Linda Corrin, Paula de Barba, and Gregor Kennedy. Visualizing patterns of student engagement and performance in moocs. In *Proceedings of the fourth international conference on learning analytics and knowledge*, pages 83–92. ACM, 2014.

[50] John Coley, Patrick Shafto, Olga Stepanova, and Elizabeth Baraff. Knowledge and category-based induction. 2005.

[51] Cristina Conati, Abigail Gertner, and Kurt Vanlehn. Using bayesian networks to manage uncertainty in student modeling. *User modeling and user-adapted interaction*, 12(4):371–417, 2002.

[52] Cristina Conati, Abigail S Gertner, Kurt VanLehn, and Marek J Druzdzel. On-line student modeling for coached problem solving using bayesian networks. In *User Modeling*, pages 231–242. Springer, 1997.

[53] Michele Conforti, Gérard Cornuéjols, Giacomo Zambelli, et al. *Integer programming*, volume 271. Springer, 2014.

[54] Albert Corbett. Cognitive computer tutors: Solving the two-sigma problem. In *International Conference on User Modeling*, pages 137–147. Springer, 2001.

[55] Albert T Corbett and John R Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4(4):253–278, 1994.

[56] Albert T Corbett, Kenneth R Koedinger, and John R Anderson. Intelligent tutoring systems. In *Handbook of human-computer interaction*, pages 849–874. Elsevier, 1997.

[57] Scott Crossley, Luc Paquette, Mihai Dascalu, Danielle S McNamara, and Ryan S Baker. Combining click-stream data with NLP tools to better understand MOOC completion. In *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge* (LAK 2016), 25–29 April 2015, Edinburgh, United Kingdom, pages 6–14, New York, USA, 2016. ACM.

[58] Sanjoy Dasgupta, Daniel Hsu, Stefanos Poulis, and Xiaojin Zhu. Teaching a black-box learner. In *International Conference on Machine Learning*, pages 1547–1555, 2019.

[59] Phil Davies. There's no confidence in multiple-choice testing,....... 2002.

[60] Chris Dede. Immersive interfaces for engagement and learning. *science*, 323(5910):66–69, 2009.

[61] Stanislas Dehaene. *How We Learn: Why Brains Learn Better Than Any Machine... for Now.* Penguin, 2020.

[62] Gerben W Dekker, Mykola Pechenizkiy, and Jan M Vleeshouwers. Predicting students drop out: A case study. In Tiffany Barnes, Michel Desmarais, Cristóbal Romero, and Sebastián Ventura, editors, *Proceedings of the Second International Conference on Educational Data Mining* (EDM '09), 1–3 July 2009, Cordoba, Spain, 2009.

[63] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):pp. 1–38, 1977.

[64] Sharon J Derry, Cindy E Hmelo-Silver, Anandi Nagarajan, Ellina Chernobilsky, and Brian D Beitzel. Cognitive transfer revisited: Can we exploit new media to solve old problems on a large scale? *Journal of Educational Computing Research*, 35(2):145–162, 2006.

[65] Nicholas Diana, John Stamper, and Ken Koedinger. An instructional factors analysis of an online logical fallacy tutoring system. In *International Conference on Artificial Intelligence in Education*, pages 86–97. Springer, 2018.

[66] Pierre Dillenbourg. Design for classroom orchestration. *Computers & Education*, 69:485–492, 2013.

[67] Pierre Dillenbourg. *Orchestration Graphs*. EPFL Press, Lausanne, Switzerland, 2015.

[68] Pierre Dillenbourg, Sanna Järvelä, and Frank Fischer. The evolution of research on computer-supported collaborative learning. In *Technology-enhanced learning*, pages 3–19. Springer, 2009.

[69] Pierre Dillenbourg, Nan Li, and Lukasz Kidzinski. The Complications of the Orchestration Clock. Technical report, Portland Press, 2016.

[70] Pierre Dillenbourg, Guillaume Zufferey, Hamed Seyed Alavi, Patrick Jermann, Lenh Hung Son Do, Quentin Bonnard, Sébastien Cuendet, and Frédéric Kaplan. Classroom orchestration: The third circle of usability. In *Proceedings of the Ninth Annual Conference on Computer-Supported Collaborative Learning* (CSCL2011), 4–8 July 2011, Hong Kong, China, volume 1, pages 510–517. International Society of the Learning Sciences, 2011.

[71] Dante D Dixson and Frank C Worrell. Formative and summative assessment in the classroom. *Theory into practice*, 55(2):153–159, 2016.

[72] Igor Douven. Scoring in context. *Synthese*, pages 1–16, 2018.

[73] Sidney K D'mello, Scotty D Craig, Amy Witherspoon, Bethany Mcdaniel, and Arthur Graesser. Automatic detection of learner's affect from conversational cues. *User Modeling and User-Adapted Interaction*, 18(1-2):45–80, 2008.

[74] Johannes C Eichstaedt, Robert J Smith, Raina M Merchant, Lyle H Ungar, Patrick Crutchley, Daniel Preoţiuc-Pietro, David A Asch, and H Andrew Schwartz. Facebook language predicts depression in medical records. *Proceedings of the National Academy of Sciences*, 115(44):11203–11208, 2018.

[75] Andrew J Elliot and Todd M Thrash. Approach-avoidance motivation in personality: approach and avoidance temperaments and goals. *Journal of personality and social psychology*, 82(5):804, 2002.

[76] Louis Faucon and Pierre Dillenbourg. Multi-armed bandits for addressing the exploration/exploitation trade-off in self improving learning environment. 2017.

[77] Louis Faucon, Lukasz Kidzinski, and Pierre Dillenbourg. Semi-markov model for simulating mooc students. *International Educational Data Mining Society*, 2016.

[78] Louis Faucon, Jennifer K Olsen, and Pierre Dillenbourg. A bayesian model of individual differences and flexibility in inductive reasoning for categorization of examples. In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, pages 285–294, 2020.

[79] Louis Faucon, Jennifer Kaitlyn Olsen, Stian Haklev, and Pierre Dillenbourg. Real-time prediction of students' activity progress and completion rates. *Journal of Learning Analytics*, 7(2):18–44, 2020.

[80] Crícia Z Felício, Klérisson VR Paixão, Celia AZ Barcelos, and Philippe Preux. A multi-armed bandit model selection for cold-start user recommendation. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*, pages 32–40, 2017.

[81] Mingyu Feng, Neil Heffernan, and Kenneth Koedinger. Addressing the assessment challenge with an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction*, 19(3):243–266, 2009.

[82] Jeremy Fiel, Kimberly A Lawless, and Scott W Brown. Timing matters: Approaches for measuring and visualizing behaviours of timing and spacing of work in self-paced online teacher professional development courses. *Journal of Learning Analytics*, 5(1):25–40, 2018.

[83] Bruno de Finetti. Methods for discriminating levels of partial knowledge concerning a test item. *British Journal of Mathematical and Statistical Psychology*, 18(1):87–123, 1965.

[84] Barry J Fraser, Herbert J Walberg, Wayne W Welch, and John A Hattie. Syntheses of educational productivity research. *International Journal of Educational Research*, 11(2):147–252, 1987.

[85] Bela Frigyik, Maya Gupta, and Yihua Chen. Shadow dirichlet for restricted probability modeling. In *Advances in Neural Information Processing Systems*, pages 613–621, 2010.

[86] Abigail S Gertner, Cristina Conati, and Kurt VanLehn. Procedural help in andes: Generating hints using a bayesian network student model. *Aaai/Iaai*, 1998:106–11, 1998.

[87] Maribeth Gettinger and Jill K Seibert. Best practices in increasing academic learning time. *Best Practices in School Psychology*, IV:773–787, 2002.

[88] John C Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society: Series B (Methodological)*, 41(2):148–164, 1979.

[89] Yue Gong, Joseph E Beck, and Neil T Heffernan. Comparing knowledge tracing and performance factor analysis by using multiple model fitting procedures. In *International conference on intelligent tutoring systems*, pages 35–44. Springer, 2010.

[90] José P González-Brenes and Yun Huang. Using data from real and simulated learners to evaluate adaptive tutoring systems. In *Proceedings of the Workshops at the 18th International Conference on Artificial Intelligence in Education AIED*, 2015.

[91] James A Gordon, William M Wilkerson, David Williamson Shaffer, and Elizabeth G Armstrong. "practicing" medicine without risk: students' and educators' responses to high-fidelity patient simulation. *Academic Medicine*, 76(5):469–472, 2001.

[92] Stian Håklev, Louis Faucon, Thanasis Hadzilacos, and Pierre Dillenbourg. Orchestration graphs: Enabling rich social pedagogical scenarios in MOOCs. In *Proceedings of the Fourth ACM Conference on Learning@Scale* (L@S 2017), 20–21 April 2017, Cambridge, Massachusetts, USA, pages 261–264, New York, USA, 2017. ACM.

[93] Stian Haklev, Louis Pierre Faucon, Thanasis Hadzilacos, and Pierre Dillenbourg. Frog: rapid prototyping of collaborative learning scenarios. In *EC-TEL*, number CONF, 2017.

[94] JHM Hamers, E De Koning, and K Sijtsma. Inductive reasoning in third grade: Intervention promises and constraints. *Contemporary Educational Psychology*, 23(2):132–148, 1998.

[95] Idit Ed Harel and Seymour Ed Papert. *Constructionism.* Ablex Publishing, 1991.

[96] Rossi A Hassad. A foundation for inductive reasoning in harnessing the potential of big data. *Statistics Education Research Journal*, 19(1), 2020.

[97] Brett K Hayes and Ben R Newell. Induction with uncertain categories: When do people consider the category alternatives? *Memory & Cognition*, 37(6):730–743, 2009.

[98] Evan Heit. Features of similarity and category-based induction. In *An Interdisciplinary Workshop On Similarity And Categorisation (SimCat)*. Edinburgh, UK, 1997.

[99] Evan Heit. A bayesian analysis of some forms of inductive reasoning. *Rational models of cognition*, pages 248–274, 1998.

[100] Jorgen Hilden, J Dik F Habbema, and Beth Bjerregaard. The measurement of performance in probabilistic diagnosis. *Methods of information in medicine*, 17(04):227–237, 1978.

[101] Jeffrey F Hine, Scott P Ardoin, and Tori E Foster. Decreasing transition times in elementary school classrooms: Using computer-assisted instruction to automate intervention components. *Journal of Applied Behavior Analysis*, 48(3):495–510, 2015.

[102] Cindy E Hmelo-Silver. Problem-based learning: What and how do students learn? *Educational psychology review*, 16(3):235–266, 2004.

[103] Jesse Hoey, Pascal Poupart, Craig Boutilier, and Alex Mihailidis. Pomdp models for assistive technology. In *Proc. AAAI Fall Symposium on Caring Machines: AI in Eldercare*, 2005.

[104] Natasha G Holmes, James Day, Anthony HK Park, DA Bonn, and Ido Roll. Making the failure more productive: scaffolding the invention process to improve inquiry behaviors and outcomes in invention activities. *Instructional Science*, 42(4):523–538, 2014.

[105] Deanna Hood, Séverin Lemaignan, and Pierre Dillenbourg. When children teach a robot to write: An autonomous teachable humanoid which uses simulated handwriting. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction*, pages 83–90, 2015.

[106] H Ulrich Hoppe. Computational methods for the analysis of learning and knowledge building communities. *Handbook of learning analytics*, pages 23–33, 2017.

[107] Ahmed Hosny, Chintan Parmar, John Quackenbush, Lawrence H Schwartz, and Hugo JWL Aerts. Artificial intelligence in radiology. *Nature Reviews Cancer*, 18(8):500–510, 2018.

[108] Stephen Hutt, Caitlin Mills, Nigel Bosch, Kristina Krasich, James Brockmole, and Sidney D'Mello. Out of the fr-eye-ing pan: Towards gaze-based models of attention during learning with technology in the classroom. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization* (UMAP 2017), 9–12 July 2017, Bratislava, Slovakia, pages 94–103, New York, USA, 2017. ACM.

[109] Lars Magnus Hvattum, Arne Løkketangen, and Fred Glover. Comparisons of commercial mip solvers and an adaptive memory (tabu search) procedure for a class of 0-1 integer programming problems. *Algorithmic Operations Research*, 7(1):13–20, 2012.

[110] Suhang Jiang, Adrienne Williams, Katerina Schenke, Mark Warschauer, and Diane O'dowd. Predicting mooc performance with week 1 behavior. In *Educational Data Mining 2014*, 2014.

[111] Athanassios Jimoyiannis and Vassilis Komis. Computer simulations in physics teaching and learning: a case study on students' understanding of trajectory motion. *Computers & education*, 36(2):183–204, 2001.

[112] Jelena Jovanović, Dragan Gašević, Shane Dawson, Abelardo Pardo, and Negin Mirriahi. Learning analytics to unveil learning strategies in a flipped classroom. *The Internet and Higher Education*, 33(4):74–85, 2017.

[113] Daniel Kahneman. *Thinking, fast and slow.* Macmillan, 2011.

[114] Manu Kapur and Katerine Bielaczyc. Designing for productive failure. *Journal of the Learning Sciences*, 21(1):45–83, 2012.

[115] Nancy Karweit and Robert E Slavin. Measurement and modeling choices in studies of time and learning. *American Educational Research Journal*, 18(2):157–171, 1981.

[116] Tanja Käser, Severin Klingler, Alexander G Schwing, and Markus Gross. Dynamic bayesian networks for student modeling. *IEEE Transactions on Learning Technologies*, 10(4):450–462, 2017.

[117] Tanja Käser, Severin Klingler, Alexander Gerhard Schwing, and Markus Gross. Beyond knowledge tracing: Modeling skill topologies with bayesian networks. In *International Conference on Intelligent Tutoring Systems*, pages 188–198. Springer, 2014.

[118] Dietmar K Kennepohl. Using computer simulations to supplement teaching laboratories in chemistry for distance delivery. 2001.

[119] Janesh K. Gupta Khalid S. Khan, David A. Davies. Formative self-assessment using multiple true-false questions on the internet: feedback according to confidence about correct knowledge. *Medical Teacher*, 23(2):158–163, 2001.

[120] Iftikhar Ahmed Khan, Mehreen Iftikhar, Syed Sajid Hussain, Attiqa Rehman, Nosheen Gul, Waqas Jadoon, and Babar Nazir. Redesign and validation of a computer programming course using inductive teaching method. *Plos one*, 15(6):e0233716, 2020.

[121] Samia Khan. New pedagogies on teaching science with computer simulations. *Journal of Science Education and Technology*, 20(3):215–232, 2011.

[122] René F Kizilcec, Chris Piech, and Emily Schneider. Deconstructing disengagement: analyzing learner subpopulations in massive open online courses. In *Proceedings of the third international conference on learning analytics and knowledge*, pages 170–179. ACM, 2013.

[123] Joris Klerkx, Katrien Verbert, and Erik Duval. Learning analytics dashboards. *Handbook of Learning Analytics*, 1:143–150, 2017.

[124] Kenneth R Koedinger, Noboru Matsuda, Christopher J MacLellan, and Elizabeth A McLaughlin. Methods for evaluating simulated learners: Examples from simstudent. *17th International Conference on Artificial Intelligence in Education AIED*, 5:45–54, 2015.

[125] Roger A Koehler. Overconfidence on probabilistic tests. *Journal of Educational Measurement*, 11(2):101–108, 1974.

[126] Ron Kohavi and Roger Longbotham. Online controlled experiments and a/b testing. *Encyclopedia of machine learning and data mining*, 7(8):922–929, 2017.

[127] Ingo Kollar and Frank Fischer. Orchestration is nothing without conducting—But arranging ties the two together!: A response to Dillenbourg (2011). *Computers & Education*, 69:507–509, 2013.

[128] Hanna Kurniawati, David Hsu, and Wee Sun Lee. Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces. In *Robotics: Science and systems*, volume 2008. Zurich, Switzerland., 2008.

[129] Felicitas-Maria Lahner, Andrea Carolin Lörwald, Daniel Bauer, Zineb Miriam Nouns, René Krebs, Sissel Guttormsen, Martin R Fischer, and Sören Huwendiek. Multiple true–false items: a comparison of scoring algorithms. *Advances in health sciences education*, 23(3):455–463, 2018.

[130] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[131] David Edgar K Lelei and Gordon McCalla. Exploring the issues in simulating a semi-structured learning environment: the simgrad doctoral program design.

[132] Nan Li, Łukasz Kidziński, Patrick Jermann, and Pierre Dillenbourg. How do in-video interactions reflect perceived video difficulty? In *Proceedings of the European MOOCs Stakeholder Summit 2015*, number EPFL-CONF-207968, pages 112–121. PAU Education, 2015.

[133] Nan Li, Łukasz Kidziński, Patrick Jermann, and Pierre Dillenbourg. Mooc video interaction patterns: What do they tell us? In *Design for Teaching and Learning in a Networked World*, pages 197–210. Springer International Publishing, 2015.

[134] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. Facing the cold start problem in recommender systems. *Expert Systems with Applications*, 41(4):2065–2073, 2014.

[135] Robert V Lindsey, Mohammad Khajah, and Michael C Mozer. Automatic discovery of cognitive skills to improve the prediction of student learning. In *Advances in neural information processing systems*, pages 1386–1394, 2014.

[136] Weiyang Liu, Bo Dai, Ahmad Humayun, Charlene Tay, Chen Yu, Linda B Smith, James M Rehg, and Le Song. Iterative machine teaching. *arXiv preprint arXiv:1705.10470*, 2017.

[137] Weiyang Liu, Bo Dai, Xingguo Li, Zhen Liu, James Rehg, and Le Song. Towards black-box iterative machine teaching. In *International Conference on Machine Learning*, pages 3141–3149, 2018.

[138] Yun-En Liu. *Building behavioral experimentation engines.* PhD thesis, 2015.

[139] Yun-En Liu, Travis Mandel, Emma Brunskill, and Zoran Popovic. Trading off scientific knowledge and user learning with multi-armed bandits. In *EDM*, pages 161–168, 2014.

[140] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[141] Tomasz D Loboda, Julio Guerra, Roya Hosseini, and Peter Brusilovsky. Mastery grids: An open source social educational progress visualization. In *Proceedings of the Ninth European Conference on Technology Enhanced Learning* (EC-TEL 2014), 16–19 September 2014, Graz, Austria, pages 235–248. Springer, 2014.

[142] Jason L Loeppky, Jerome Sacks, and William J Welch. Choosing the sample size of a computer experiment: A practical guide. *Technometrics*, 2012.

[143] Katharina Loibl, Ido Roll, and Nikol Rummel. Towards a theory of when and how problem solving followed by instruction supports learning. *Educational Psychology Review*, 29(4):693–715, 2017.

[144] J Derek Lomas, Jodi Forlizzi, Nikhil Poonwala, Nirmal Patel, Sharan Shodhan, Kishan Patel, Ken Koedinger, and Emma Brunskill. Interface design optimization as a multi-armed bandit problem. In *Proceedings of the 2016 CHI conference on human factors in computing systems*, pages 4142–4153, 2016.

[145] Zachary MacHardy and Zachary A Pardos. Toward the evaluation of educational videos using bayesian knowledge tracing and big data. In *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*, pages 347–350. ACM, 2015.

[146] Christopher J Maclellan, Erik Harpstead, Rony Patel, and Kenneth R Koedinger. The apprentice learner architecture: Closing the loop between learning theory and educational data. *International Educational Data Mining Society*, 2016.

[147] Christopher J MacLellan, Kenneth R Koedinger, and Noboru Matsuda. Authoring tutors with simstudent: An evaluation of efficiency and model quality. In *International conference on intelligent tutoring systems*, pages 551–560. Springer, 2014.

[148] Thomas W Malone. Toward a theory of intrinsically motivating instruction. *Cognitive Science*, 5(4):333–369, 1981.

[149] Indu Manickam, Andrew S Lan, and Richard G Baraniuk. Contextual multi-armed bandit algorithms for personalized learning action selection. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6344–6348. IEEE, 2017.

[150] Ye Mao. Deep learning vs. bayesian knowledge tracing: Student models for interventions. *Journal of educational data mining*, 10(2), 2018.

[151] Roberto Martinez-Maldonado, Andrew Clayphan, and Judy Kay. Deploying and visualising teacher's scripts of small group activities in a multi-surface classroom ecology: A study in-the-wild. *Computer Supported Cooperative Work (CSCW)*, 24(2-3):177–221, 2015.

[152] Karen D Mattingly, Margaret C Rice, and Zane L Berge. Learning analytics as a tool for closing the assessment loop in higher education. *Knowledge Management & E-Learning: An International Journal*, 4(3):236–247, 2012.

[153] Michael Mayo and Antonija Mitrovic. Optimising its behaviour with bayesian networks and decision theory. 2001.

[154] Matthew D McLure, Scott E Friedman, and Kenneth D Forbus. Learning concepts from sketches via analogical generalization and near-misses. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 32, 2010.

[155] Douglas L Medin, John D Coley, Gert Storms, and Brett L Hayes. A relevance theory of induction. *Psychonomic Bulletin & Review*, 10(3):517–532, 2003.

[156] Eva Millán, Tomasz Loboda, and Jose Luis Pérez-De-La-Cruz. Bayesian networks for student model engineering. *Computers & Education*, 55(4):1663–1683, 2010.

[157] Eva Millán and José Luis Pérez-De-La-Cruz. A bayesian diagnostic algorithm for student modeling and its evaluation. *User Modeling and User-Adapted Interaction*, 12(2-3):281–330, 2002.

[158] Eva Millán, José Luis Pérez-de-la Cruz, and Eva Suárez. Adaptive bayesian networks for multilevel student modelling. In *International Conference on Intelligent Tutoring Systems*, pages 534–543. Springer, 2000.

[159] Andrew J Milson and Brian D Earle. Internet-based gis in an inductive learning environment: A case study of ninth-grade geography students. *Journal of geography*, 106(6):227–237, 2008.

[160] Norma Ming and Vivienne Ming. Predicting student outcomes from unstructured data. In *Workshop and Poster Proceedings of the 20th Conference on User Modeling, Adaptation, and Personalization* (UMAP 2012), 16–20 July 2012, Montreal, Canada, 2012.

[161] SA22163801092 MirHassani. A computational approach to enhancing course timetabling with integer programming. *Applied Mathematics and Computation*, 175(1):814–822, 2006.

[162] Inge Molenaar and Carolien Knoop-van Campen. Teacher dashboards in practice: Usage and impact. In *Proceedings of the 12th European Conference on Technology Enhanced Learning* (EC-TEL 2017), 12–15 September 2017, Tallinn, Estonia, pages 125–138. Springer, 2017.

[163] Todd K Moon. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60, 1996.

[164] Jauwairia Nasir, Utku Norman, Wafa Johal, Jennifer K Olsen, Sina Shahmoradi, and Pierre Dillenbourg. Robot analytics: What do human-robot interaction traces tell us about learning? In *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 1–7. IEEE, 2019.

[165] Andrew Y Ng and Michael I Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in neural information processing systems*, pages 841–848, 2002.

[166] Richard Nock and Frank Nielsen. On weighting clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(8):1223–1235, 2006.

[167] Jennifer K Olsen, Louis Faucon, and Pierre Dillenbourg. Transferring interactive activities in large lectures from face-to-face to online settings. *Information and Learning Sciences*, 2020.

[168] Daniel N Osherson, Edward E Smith, Ormond Wilkie, Alejandro Lopez, and Eldar Shafir. Category-based induction. *Psychological review*, 97(2):185, 1990.

[169] Korinn Ostrow, Christopher Donnelly, Seth Adjei, and Neil Heffernan. Improving student modeling through partial credit and problem difficulty. In *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*, pages 11–20. ACM, 2015.

[170] Ayberk Özgür, Séverin Lemaignan, Wafa Johal, Maria Beltran, Manon Briod, Léa Pereyre, Francesco Mondada, and Pierre Dillenbourg. Cellulo: Versatile handheld robots for education. In *2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI*, pages 119–127. IEEE, 2017.

[171] John F Pane, Beth Ann Griffin, Daniel F McCaffrey, and Rita Karam. Effectiveness of cognitive tutor algebra i at scale. *Educational Evaluation and Policy Analysis*, 36(2):127–144, 2014.

[172] Zachary Pardos and Neil Heffernan. Navigating the parameter space of bayesian knowledge tracing models: Visualizations of the convergence of the expectation maximization algorithm. In *Educational Data Mining 2010*, 2010.

[173] Zachary A Pardos, Yoav Bergner, Daniel T Seaton, and David E Pritchard. Adapting bayesian knowledge tracing to a massive open online course in edx. In *EDM*, pages 137–144, 2013.

[174] Zachary A Pardos and Neil T Heffernan. Modeling individualization in a bayesian networks implementation of knowledge tracing. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 255–266. Springer, 2010.

[175] Zachary A Pardos and Neil T Heffernan. Kt-idem: Introducing item difficulty to the knowledge tracing model. In *International conference on user modeling, adaptation, and personalization*, pages 243–254. Springer, 2011.

[176] Philip I Pavlik Jr., Hao Cen, and Kenneth R Koedinger. Performance factors analysis—A new alternative to knowledge tracing. 2009. online submission.

[177] Radek Pelánek. Metrics for evaluation of student models. *Journal of Educational Data Mining*, 7(2):1–19, 2015.

[178] Amy Perfors, Joshua B Tenenbaum, Thomas L Griffiths, and Fei Xu. A tutorial introduction to bayesian models of cognitive development. *Cognition*, 120(3):302–321, 2011.

[179] Lighton Phiri, Christoph Meinel, and Hussein Suleman. Streamlined orchestration: An orchestration workbench framework for effective teaching. *Computers & Education*, 95:231–238, 2016.

[180] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein. Deep knowledge tracing. In *Advances in neural information processing systems*, pages 505–513, 2015.

[181] Sharon B Poizner, W Alan Nicewander, and Charles F Gettys. Alternative response and scoring methods for multiple-choice items: An empirical study of probabilistic and ordinal response modes. *Applied Psychological Measurement*, 2(1):83–96, 1978.

[182] Martha C Polson and J Jeffrey Richardson. *Foundations of intelligent tutoring systems*. Psychology Press, 2013.

[183] Luis P Prieto, Martina Holenko Dlab, Israel Gutiérrez, Mahmoud Abdulwahed, and Walid Balid. Orchestrating technology enhanced learning: A literature review and a conceptual framework. *International Journal of Technology Enhanced Learning*, 3(6):583, 2011.

[184] Luis Pablo Prieto, Juan Ignacio Asensio-Pérez, Yannis Dimitriadis, Eduardo Gómez-Sánchez, and Juan Alberto Muñoz-Cristóbal. Glue!-ps: a multi-language architecture and data model to deploy tel designs to multiple learning environments. In *European Conference on Technology Enhanced Learning*, pages 285–298. Springer, 2011.

[185] Michael J Prince and Richard M Felder. Inductive teaching and learning methods: Definitions, comparisons, and research bases. *Journal of engineering education*, 95(2):123–138, 2006.

[186] Shi Pu, Michael Yudelson, Lu Ou, and Yuchi Huang. Deep knowledge tracing with transformers. In *International Conference on Artificial Intelligence in Education*, pages 252–256. Springer, 2020.

[187] Yumeng Qiu, Yingmei Qi, Hanyuan Lu, Zachary A Pardos, and Neil T Heffernan. Does time matter? modeling the effect of time with bayesian knowledge tracing. In *EDM*, pages 139–148, 2011.

[188] Mirko Raca, Łukasz Kidziński, and Pierre Dillenbourg. Translating head motion into attention-towards processing of student's body-language. In *Proceedings of the 8th International Conference on Educational Data Mining*, number EPFL-CONF-207803, 2015.

[189] Anna Rafferty, Huiji Ying, and Joseph Williams. Statistical consequences of using multi-armed bandits to conduct adaptive educational experiments. *JEDM| Journal of Educational Data Mining*, 11(1):47–79, 2019.

[190] Anna N Rafferty, Emma Brunskill, Thomas L Griffiths, and Patrick Shafto. Faster teaching via pomdp planning. *Cognitive science*, 40(6):1290–1332, 2016.

[191] Anna N Rafferty, Huiji Ying, and Joseph Jay Williams. Bandit assignment for educational experiments: Benefits to students versus statistical power. In *International Conference on Artificial Intelligence in Education*, pages 286–290. Springer, 2018.

[192] Howard Raiffa and Robert Schlaifer. Applied statistical decision theory. 1961.

[193] Arti Ramesh, Dan Goldwasser, Bert Huang, Hal Daumé III, and Lise Getoor. Modeling learner engagement in moocs using probabilistic soft logic. In *NIPS Workshop on Data Driven Education*, 2013.

[194] Katrina N Rhymer, Christopher H Skinner, Shantwania Jackson, Stephanie McNeill, Tawnya Smith, and Bertha Jackson. The 1-minute explicit timing intervention: The influence of mathematics problem difficulty. *Journal of Instructional Psychology*, 29(4):305–311, 2002.

[195] Robert Rippey. Probabilistic testing. *Journal of Educational Measurement*, 5(3):211–215, 1968.

[196] Robert M Rippey. A comparison of five different scoring functions for confidence tests 1. *Journal of Educational Measurement*, 7(3):165–170, 1970.

[197] Kelly Rivers, Erik Harpstead, and Kenneth R Koedinger. Learning curve analysis for programming: Which concepts do students struggle with? In *ICER*, pages 143–151, 2016.

[198] Henry L Roediger III, Pooja K Agarwal, Mark A McDaniel, and Kathleen B McDermott. Test-enhanced learning in the classroom: long-term improvements from quizzing. *Journal of Experimental Psychology: Applied*, 17(4):382, 2011.

[199] Ido Roll, Daniel M Russell, and Dragan Gašević. Learning at scale. *International Journal of Artificial Intelligence in Education*, 28(4):471–477, 2018.

[200] Stéphane Ross, Joelle Pineau, Sébastien Paquet, and Brahim Chaib-Draa. Online planning algorithms for pomdps. *Journal of Artificial Intelligence Research*, 32:663–704, 2008.

[201] Stuart Russell and Peter Norvig. Artificial intelligence: a modern approach. 2002.

[202] Nico Rutten, Jan T van der Veen, and Wouter R van Joolingen. Inquiry-based whole-class teaching with computer simulations in physics. *International journal of science education*, 37(8):1225–1245, 2015.

[203] Ruslan Salakhutdinov, Joshua Tenenbaum, and Antonio Torralba. One-shot learning with a hierarchical nonparametric bayesian model. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, pages 195–206, 2012.

[204] Daniel L Schwartz and John D Bransford. A time for telling. *Cognition and instruction*, 16(4):475–5223, 1998.

[205] Daniel L Schwartz and Taylor Martin. Inventing to prepare for future learning: The hidden efficiency of encouraging original student production in statistics instruction. *Cognition and Instruction*, 22(2):129–184, 2004.

[206] Beat A Schwendimann, Maria Jesus Rodriguez-Triana, Andrii Vozniuk, Luis P Prieto, Mina Shirvani Boroujeni, Adrian Holzer, Denis Gillet, and Pierre Dillenbourg. Perceiving learning at a glance: A systematic literature review of learning dashboard research. *IEEE Transactions on Learning Technologies*, 10(1):30–41, 2016.

[207] Steven L Scott. A modern bayesian look at the multi-armed bandit. *Applied Stochastic Models in Business and Industry*, 26(6):639–658, 2010.

[208] Yvonne Sedelmaier and Dieter Landes. Active and inductive learning in software engineering education. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, volume 2, pages 418–427. IEEE, 2015.

[209] Avi Segal, Yossi Ben David, Joseph Jay Williams, Kobi Gal, and Yaar Shalom. Combining difficulty ranking with multi-armed bandits to sequence educational content. In *International conference on artificial intelligence in education*, pages 317–321. Springer, 2018.

[210] John Self. The defining characteristics of intelligent tutoring systems research: Itss care, precisely. 1998.

[211] John A Self. Student models in computer-aided instruction. *International Journal of Man-machine studies*, 6(2):261–276, 1974.

[212] Bar Shalem, Yoram Bachrach, John Guiver, and Christopher M Bishop. Students, teachers, exams and moocs: Predicting and optimizing attainment in web-based education using a probabilistic graphical model. In *Machine Learning and Knowledge Discovery in Databases*, pages 82–97. Springer, 2014.

[213] Kshitij Sharma, Patrick Jermann, and Pierre Dillenbourg. “with-me-ness”: A gaze-measure for students’ attention in moocs. In *Proceedings of international conference of the learning sciences 2014*, number CONF, pages 1017–1022. ISLS, 2014.

[214] Kshitij Sharma, Zacharoula Papamitsiou, Jennifer K Olsen, and Michail Giannakos. Predicting learners’ effortful behaviour in adaptive assessment using multimodal data. In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, pages 480–489, 2020.

[215] Jonathan T Shemwell, Catherine C Chase, and Daniel L Schwartz. Seeking the general explanation: A test of inductive activities for learning and transfer. *Journal of Research in Science Teaching*, 52(1):58–83, 2015.

[216] Emir H Shuford, Arthur Albert, and H Edward Massengill. Admissible probability measurement procedures. *Psychometrika*, 31(2):125–145, 1966.

[217] Valerie Shute and Brendon Towle. Adaptive e-learning. *Educational psychologist*, 38(2):105–114, 2003.

[218] Valerie J Shute. A comparison of learning environments: All that glitters. *Computers as cognitive tools*, pages 47–74, 1993.

[219] Valerie J Shute and Robert Glaser. A large-scale evaluation of an intelligent discovery world: Smithtown. *Interactive learning environments*, 1(1):51–77, 1990.

[220] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[221] Stefan Slater, Srećko Joksimović, Vitomir Kovanovic, Ryan S Baker, and Dragan Gasevic. Tools for educational data mining: A review. *Journal of Educational and Behavioral Statistics*, 42(1):85–106, 2017.

[222] James D Slotta, Mike Tissenbaum, and Michelle Lui. Orchestrating of complex inquiry: Three roles for learning analytics in a smart classroom infrastructure. In *Proceedings of the Third International Conference on Learning Analytics and Knowledge* (LAK 2013), 8–12 April 2013, Leuven, Belgium, pages 270–274, New York, USA, 2013. ACM.

[223] Karl L Smart, Christine Witt, and James P Scott. Toward learner-centered teaching: An inductive approach. *Business Communication Quarterly*, 75(4):392–403, 2012.

[224] Lara Kathleen Smetana and Randy L Bell. Computer simulations to support science instruction and learning: A critical review of the literature. *International Journal of Science Education*, 34(9):1337–1370, 2012.

[225] Linda B Smith, Susan S Jones, Barbara Landau, Lisa Gershkoff-Stowe, and Larissa Samuelson. Object name learning provides on-the-job training for attention. *Psychological science*, 13(1):13–19, 2002.

[226] Trevor I Smith, Kyle J Louis, Bartholomew J Ricci IV, and Nasrine Bendjilali. Quantitatively ranking incorrect responses to multiple-choice questions using item response theory. *Physical Review Physics Education Research*, 16(1):010107, 2020.

[227] Ray J Solomonoff. Algorithmic probability: Theory and applications. In *Information theory and statistical learning*, pages 1–23. Springer, 2009.

[228] Ray J Solomonoff. Algorithmic probability–its discovery–its properties and application to strong ai. *Randomness Through Computation: Some Answers, More Questions*, pages 1–23, 2011.

[229] Yanjie Song, Lung-Hsiang Wong, and Chee-Kit Looi. Fostering personalized learning in science inquiry supported by mobile technologies. *Educational Technology Research and Development*, 60(4):679–701, 2012.

[230] Colin Taylor, Kalyan Veeramachaneni, and Una-May O'Reilly. Likely to stop? Predicting stopout in massive open online courses. *arXiv preprint arXiv:1408.3382*, 2014.

[231] Joshua B Tenenbaum. Bayesian modeling of human concept learning. In *Advances in neural information processing systems*, pages 59–68, 1999.

[232] Joshua B Tenenbaum, Charles Kemp, Thomas L Griffiths, and Noah D Goodman. How to grow a mind: Statistics, structure, and abstraction. *science*, 331(6022):1279–1285, 2011.

[233] William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

[234] Lawrence A Tomassini, Ira Solomon, Marshall B Romney, and Jack L Krogstad. Calibration of auditors' probabilistic judgments: Some empirical evidence. *Organizational Behavior and Human Performance*, 30(3):391–406, 1982.

[235] Amos Tversky. Features of similarity. *Psychological review*, 84(4):327, 1977.

[236] MJ Tweed, S Stein, TJ Wilkinson, G Purdie, and J Smith. Certainty and safe consequence responses provide additional information from multiple choice question assessments. *BMC medical education*, 17(1):106, 2017.

[237] Wim J van der Linden and Ronald K Hambleton. *Handbook of modern item response theory*. Springer Science & Business Media, 2013.

[238] Ronald Van Houten, Sharon Hill, and Madeline Parsons. An analysis of a performance feedback system: The effects of timing and feedback, public posting, and praise upon academic performance and peer interaction 1. *Journal of Applied Behavior Analysis*, 8(4):449–457, 1975.

[239] Ronald Van Houten and Cathy Thompson. The effects of explicit timing on math performance. *Journal of Applied Behavior Analysis*, 9(2):227, 1976.

[240] Anouschka van Leeuwen, Nikol Rummel, and Tamara van Gog. What information should CSCL teacher dashboards provide to help teachers interpret CSCL situations? *International Journal of Computer-Supported Collaborative Learning*, 14:261–289, 2019.

[241] Jef Vanderoost, Rianne Janssen, Jan Eggermont, Riet Callens, and Tinne De Laet. Elimination testing with adapted scoring reduces guessing and anxiety in multiple-choice assessments, but does not increase grade average in comparison with negative marking. *PloS one*, 13(10), 2018.

[242] Kurt Vanlehn. The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education*, 16(3):227–265, 2006.

[243] Kurt VanLehn. The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*, 46(4):197–221, 2011.

[244] Katrien Verbert, Sten Govaerts, Erik Duval, Jose Luis Santos, Frans Assche, Gonzalo Parra, and Joris Klerkx. Learning dashboards: An overview and future research opportunities. *Personal and Ubiquitous Computing*, 18(6):1499–1514, 2014.

[245] Sofía S Villar, Jack Bowden, and James Wason. Multi-armed bandit models for the optimal design of clinical trials: benefits and challenges. *Statistical science: a review journal of the Institute of Mathematical Statistics*, 30(2):199, 2015.

[246] Ipke Wachsmuth and Jens-Holger Lorenz. Sharpening one's diagnostic skill by simulating students' error behaviors. *Focus on learning problems in mathematics*, 9(2), 1987.

[247] Josef Wachtler, Mohammad Khalil, Behnam Taraghi, and Martin Ebner. On using learning analytics to track the activity of interactive MOOC videos. In *Proceedings of the LAK 2016 Workshop on Smart Environments and Analytics in Video-Based Learning*, 26 April 2016, Edinburgh, United Kingdom, pages 8–17, 2016.

[248] Patrick Wang, Pierre Tchounikine, and Matthieu Quignard. Chao: A framework for the development of orchestration technologies for technology-enhanced learning activities using tablets in classrooms. *International Journal of Technology Enhanced Learning*, 10(1/2):1–21, 2017.

[249] Yutao Wang and Neil Heffernan. Extending knowledge tracing to allow partial credit: Using continuous versus binary nodes. In *International conference on artificial intelligence in education*, pages 181–188. Springer, 2013.

[250] Yun Wen, Luis P Prieto, and Pierre Dillenbourg. Paper-based tabletop application for collaborative chinese character learning. In *11th International Conference on Computer-Supported Collaborative Learning,(c)*, pages 1–2, 2015.

[251] Jacob Whitehill, Kiran Mohan, Daniel Seaton, Yigal Rosen, and Dustin Tingley. Delving deeper into MOOC student dropout prediction. *arXiv preprint arXiv:1702.06404*, 2017.

[252] Joseph Jay Williams, Anna N Rafferty, Dustin Tingley, Andrew Ang, Walter S Lasecki, and Juho Kim. Enhancing online problems through instructor-centered tools for randomized experiments. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2018.

[253] Patrick H Winston. Learning structural descriptions from examples. 1970.

[254] Christopher A Wolters. Advancing achievement goal theory: Using goal structures and goal orientations to predict students' motivation, cognition, and achievement. *Journal of educational psychology*, 96(2):236, 2004.

[255] Dongming Xu, Huaiqing Wang, and Minhong Wang. A conceptual model of personalized virtual learning environments. *Expert Systems with Applications*, 29(3):525–534, 2005.

[256] Jamie L Yarbrough, Christopher H Skinner, Young Ju Lee, and Cathy Lemmons. Decreasing transition times in a second grade classroom: Scientific support for the timely transitions game. *Journal of Applied School Psychology*, 20(2):85–107, 2004.

[257] Teresa Yeo, Parameswaran Kamalaruban, Adish Singla, Arpit Merchant, Thibault Asselborn, Louis Faucon, Pierre Dillenbourg, and Volkan Cevher. Iterative classroom teaching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5684–5692, 2019.

[258] Michael V Yudelson, Kenneth R Koedinger, and Geoffrey J Gordon. Individualized bayesian knowledge tracing models. In *International conference on artificial intelligence in education*, pages 171–180. Springer, 2013.

[259] Jerry Zhu. Machine teaching for bayesian learners in the exponential family. In *Advances in Neural Information Processing Systems*, pages 1905–1913, 2013.

[260] Xiaojin Zhu. Machine teaching: An inverse problem to machine learning and an approach toward optimal education. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[261] Xiaojin Zhu, Adish Singla, Sandra Zilles, and Anna N Rafferty. An overview of machine teaching. *arXiv preprint arXiv:1801.05927*, 2018.

Louis Faucon

Contact

Nationality	French
Date of Birth	10 October 1991
Phone	+41 78 662 91 30
Email	lpfaucon@gmail.com

Experience

2020–current **Climate Risk Software Engineer at MSCI:** Within the Climate Risk Center in Zurich, I am developing a computation platform for research models. I focus on questions around data collection, data quality and computation performance.

2016–2020 **Contributor for the FROG project:** FROG is an open source platform for in-classroom teaching focused on collaborative learning, experiential learning and learning analytics. The platform has been used in multiple classrooms with several hundreds of students.

2015–2016 **Master Thesis at Coorpacademy** (start-up co-founded by Jean Marc Tassetto, former president of Google France). In Lausanne, Switzerland for 7 months. My role was to implement a robust and efficient data architecture and build on it tools such as analytics dashboards and recommender systems. Worked with Spark, Hadoop, Neo4J, Elasticsearch, MongoDB and AWS.

2014—2015 **Research Scholar at EPFL's laboratory CHILI** for 1 year in Lausanne. I participated in the research of a post-doctorate scientist and carried a project of my own. My project was to automatically extract categories of behavior of MOOCs students using Machine Learning. User behavior was described with a Semi-Markov Model and the algorithm was trained using Expectation-Maximisation on a dataset of 20 million events of online students from the platforms Coursera and EdX. The analysis was done in Python and lead to a scientific publication.

2014 **Research internship at LRI-INRIA** within the team In Situ on Human-Computer Interaction during 5 months in Orsay, France. I developed an Android application for remote communication and mutual awareness.

2013 **Summer internship in Digital Media Solution**, a company specialised in immersive sound. Developed an Android application which plays sound in 3D. In Noisiel, France for 6 weeks.

2011–2012 **Internship in the French armed forces**. Lasted 8 month in several parts of France. Engaged in daily military training, was promoted officer cadet and was given command of a platoon of 38 soldiers for training, teaching and a mission of security in Paris.

Education

2016–2020 **Computer Science PhD at École Polytechnique Fédérale de Lausanne**, supervised by Professor Dillenbourg on applying computational methods to educational technology. I studied how a learning environment can be adaptive (personalizing teaching to individual students) and self-improving (learning to teach better over time), with a focus on inductive teaching and inductive reasoning. To this end, I used several Machine Learning techniques, notably Bayesian networks. Additionaly the PhD program also includes teaching duties and mentoring semester-long project of Master and Bachelor students.

2014–2016 Completed the **Master in Computer Science at École Polytechnique Fédérale de Lausanne**, which is one of Europe leading university for Engineering, Computer Science, and Machine Learning.

2011–2015 I received an engineering degree from **École Polytechnique (X)** (Palaiseau, France). The French leading university for high-level scientific training. Multidisciplinary courses (mathematics, physics, mechanics, computer science, humanities) during the first two years, then a specialization in computer science.

2009–2011 Classe préparatoire at **Lycée Henri IV** (Paris). Two-year intensive program in preparation for competitive examinations to the French Grandes Écoles for scientific studies. Program in mathematics, physics and computer science. Accepted at the École Polytechnique and the École Normale Supérieure Ulm.

Projects

2021–current **Treasurer and technical contributor of the Tournesol association**: Tournesol is a project developing a data collection platform focused on the Ethics of Artificial Intelligence and large scale Recommender Systems. We seek to build a robust mechanism to collectively decide which content are beneficial for the public and should be more recommended.

2019–2020 **VP of EPFL's Effective Altruism student association**, we organized introductory workshops and discussions about the principles of EA. I also founded a reading group on Artificial Intelligence. We have met weekly since September 2019 to discuss robustness, value specification, and the impact of AI systems.

2019–2020 **Bayes-Up** is a web application for sharing Bayesian-Multiple Choice Quizzes. Such a quiz must be answered by assigning probabilities to each choice and uses an incentive compatible scoring rule. It can be used to teach probabilistic thinking and intellectual honesty. `bayes-up.web.app`

Since 2016 **Web Development**: I develop and deploy web applications as a hobby. Notably, games (set, quarto, ricochet-robots), or productivity apps (pomodoro, spaced-repetition)

Programming Skills

Python, numpy, pandas, Django JavaScript, React, Eslint, Meteor, Material-UI, Firebase, SQL, Git, Github, bash

Publications

2021 Lê Nguyen Hoang, Louis Faucon, and El-Mahdi El-Mhamdi. (2021) "Recommendation algorithms, a neglected opportunity for public health." *Revue Médecine et Philosophie*, `https://philpapers.org/rec/HOARAA`

2020 Louis Faucon, Jennifer K. Olsen, and Pierre Dillenbourg. "A Bayesian model of individual differences and flexibility in inductive reasoning for categorization of examples." (2020) *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, `https://doi.org/10.1145/3375462.3375512`

2020 Louis Faucon, Jennifer K. Olsen, Stian Haklev, and Pierre Dillenbourg. (2020) "Real-Time prediction of students' activity progress and completion rates." *Journal of Learning Analytics*, `https://doi.org/10.18608/jla.2020.72.2`

2020 Jennifer K. Olsen, Louis Faucon, and Pierre Dillenbourg. "Transferring interactive activities in large lectures from face-to-face to online settings." (2020) *Information and Learning Sciences.* `https://doi.org/10.1108/ILS-04-2020-0109`

2019 Teresa Yeo, Parameswaran Kamalaruban, Adish Singla, Arpit Merchant, Thibault Asselborn, Louis Faucon, Pierre Dillenbourg, and Volkan Cevher. "Iterative classroom teaching." (2019) In *Proceedings of the AAAI Conference on Artificial Intelligence*, `https://arxiv.org/abs/1811.03537`

2017 Stian Haklev, Louis Faucon, Thanasis Hadzilacos, and Pierre Dillenbourg. "Orchestration graphs: Enabling rich social pedagogical scenarios in MOOCs." (2017) *Proceedings of the Fourth ACM Conference on Learning@Scale.*, `https://doi.org/10.1145/3051457.3054000`

2017 Stian Haklev, Louis Faucon, Thanasis Hadzilacos, and Pierre Dillenbourg. "FROG: rapid prototyping of collaborative learning scenarios." (2017) *EC-TEL.*

2017 Louis Faucon. "Multi-Armed Bandits for Addressing the Exploration/Exploitation Trade-off in Self Improving Learning Environment."

2016 Louis Faucon, Lukasz Kidzinski, and Pierre Dillenbourg. "Semi-Markov Model for Simulating MOOC Students." *International Educational Data Mining Society* , `https://www.educationaldatamining.org/EDM2016/proceedings/paper_112.pdf`

Google Scholar Find a more detailed list of publications on my profile at `https://scholar.google.com/citations?user=0oLMF0sAAAAJ`

Awards

2015 Finalist of **Google Hashcode** programming competition

2008 **Olympiades académiques de mathématiques**: a French high-level math competition. I ranked 7th over more than a thousand competitors.

Languages

French	Native
English	Fluent
Thai	Beginner