Integrated intelligent LEARNing environment for Reading and Writing

D4.3 – Interface and game-scenario adaptation mechanisms

Information Society and Media

Document identifier	**ILearnRW_D4.3_Adaptation**
Date	**03/31/14**
WP	**WP4**
Partners	**UoM, DYSACT**
WP Lead Partner	**UoM**
Document status	**v0.9**

Deliverable Number	D4.3
Deliverable Title	Interface and game-scenarios adaptation
Deliverable version number	v0.9
Work package	WP4
Task	Task 4.3 Interface and game-scenario adaptation
Nature of the deliverable	Report (R)
Dissemination level	Public (**PU**)
Date of Version	03/31/14

Author(s)	Héctor P. Martínez
Contributor(s)	Georgios N. Yannakakis
Reviewer(s)	Daniel Gooch

Abstract	This deliverable describes the infrastructure that the game relies on in order to provide content that is tailored to the needs of each individual student. Part of this infrastructure has been described in previous deliverables, thus the focus is on the *lesson planner* component that provides an additional layer of adaptation over the student model. The methods utilised to create this component, namely a crowd-sourcing survey coupled with a data-driven modelling method, are described in detail.

Keywords	Game adaptation, personalization, preference learning, decision trees, data-driven model building

Document Status Sheet

Issue	Date	Comment	Author
v0.1	02/27/2014	Formatting and initial related work	Héctor P. Martínez
v0.2	03/07/2014	Related work + intro	Héctor P. Martínez
v0.3	03/11/2014	Crowdsourcing + decision trees	Héctor P. Martínez
v0.4	03/23/2014	First draft	Héctor P. Martínez
v0.5	03/24/2014	First internal review	Georgios N. Yannakakis
v0.6	03/25/2014	Second internal review	Daniel Gooch
v0.7	03/26/2014	Final revision	Héctor P. Martínez
v0.8	03/28/2014	Final formatting	Héctor P. Martínez
v0.9	03/31/2014	Proofreading	Georgios N. Yannakakis

Project information

Project acronym:	**ILearnRW**
Project full title:	**Integrated Intelligent Learning Environment for Reading and Writing**
Proposal/Contract no.:	**318803**

Project Officer: Krister Olson

Address:	L-2920 Luxembourg, Luxembourg
Phone:	+35 2430 134 332
E-mail:	krister.olson@ec.europa.eu

Project Co-ordinator: Noel Duffy

Address:	Dolphin Computer Access Ltd. Technology House, Blackpole Estate West, Worcester, UK. WR3 8TJ
Phone:	+01 905 754 577
Fax:	+01 905 754 559
E-mail:	noel.duffy@yourdolphin.com

Table of Content

1. Introduction

This project takes advantage of the qualities of games in order to create a learning tool that will keep students engaged while working on their reading and writing skills within and outside the classroom. While interesting mechanics and attractive audiovisual content are essential for keeping the students interested, these are not sufficient to offer an effective learning experience for any student. Adaptation mechanisms are fundamental to make the game accessible and effective for a broad variety of students, each with her own particular needs and set of skills. This deliverable describes the different components and methods developed within the iLearnRW system that contribute to the personalisation and adaptation of the game.

Using adaptation in games is not a new idea, in fact adaptation has attracted significant interest in game industry and academia, often aimed at enhancing engagement or enjoyment by tailoring the level of challenge[1] to individual players (see e.g. (Spronck, Sprinkhuizen-Kuyper & Postma, 2004) among many others). In games for learning (or as matter of fact, any form of learning environment), while engagement is sometimes used, the ultimate aim of adaptation mechanisms is the achievement of a particular set of learning outcomes and the optimisation of the path towards achieving these goals.

Games present numerous facets that contribute to better learning outcomes, but there are four elements in particular that are *key* towards achieving a personalised and effective learning experience. First of all, the game has to deliver the learning content (i.e. *curriculum*) in a sensible order (e.g. in a learning game about dyslexia the drop suffixing rule used in "making" should not be presented before basic "-ing" suffixing). Second, the game should also take into account the student's knowledge and skills to decide which content to teach next (e.g. if the student has not yet understood basic suffixing, the drop rule should not be presented yet). Third, within one playing session, the system should be able to combine activities and content that keep the student motivated and engaged (e.g. if the student is having difficulties learning the drop rule, then the system might inject activities for basic suffixing that the student already knows, thus increasing her moral). Forth and last, the content presentation should match the student preferences and skills (e.g. the speed of the text should not be too fast or too slow for the reading level of the student).

In the field of intelligent tutoring systems (ITS), these four elements are often controlled by separate but dependent components referred to as the *domain knowledge*, *student model*, *pedagogical*, and *communication* modules, respectively (Beck, Stern, & Haugsjaa, 1996). In the iLearnRW system, the first two are already supported by the user profile (see User modelling deliverable, D4.1) while the communication module is integrated on the game development through the realisation of the teaching strategies (e.g. multi-sensory approach) and the text presentation services deployed in the server and described in deliverable D3.5. This deliverable, instead, focuses on the remaining module, the **pedagogical adaptation** module, that given the ideal order of the curriculum (i.e. specific difficulties), the literacy skills of the student, and her performance on the game, plans an optimal game session (or

[1] In this deliverable we reserve the word *difficulty* for the *specific literacy difficulties* collected in the user profile, while we use the word *challenge* for the general level of difficulty in a game.

lesson plan) that balances the goals of advancing in the curriculum, reinforcing already acquired concepts, and keeping the student engaged and motivated.

In related work, this module is often hardcoded and in many cases considered trivial. However, the complexity involved in teaching adequately reading and writing to young students with dyslexia urges for a more careful and bottom-up design. Consider for instance the strong emphasis that it is usually given to *overlearning* (i.e. work on the same difficulty repeatedly in order to secure the learned concepts) in contraposition to the certain detriment of student interest when playing over and over the same activity. On the other end, when working too often on new difficulties the student might find herself losing at the game frequently which can be a serious factor for her (potential lack of) motivation and confidence. While a well-designed game may ease the challenges associated to interest and motivation, we believe that including a pedagogical module to a game-based learning software is necessary for delivering an effective learning experience.

While there exist general teaching guidelines on how to structure a dyslexia intervention, practical wisdom is still an important part of the process. This has greatly limited the capabilities of current tools which cannot be used effectively without a close teacher supervision. Following the principles of participatory design used in other parts of the project, we create a pedagogical module by trying to capture this practical wisdom from a large group of teachers specialised on dyslexia. Instead of organising a workshop session, we decided to set up a data-driven approach (via crowdsourcing) that would allow us to harness the knowledge of as many experts as possible, and machine-learn the techniques which are more frequently used. We argue that a data-driven approach via online surveys offers an alternative and possibly more suitable option for this task as it allow us (for the first time) to reach a much larger number of participants, and thereby obtain collective knowledge from a large set of experts on the matter (SEN teachers).

In order to extract as much unbiased information from the survey as possible, we propose a computational method to find regularities within the responses. In particular, we propose an entirely new algorithm based on *decision trees* that can learn from annotations/answers given through rank-based questionnaires. The great advantage of using the ***decision tree preference learning*** method is that it provides a highly expressive and human-readable computational model that can be validated by the experts in our consortium and, in general, provide an insight on the strategies used by experts to plan dyslexia interventions.

In sum, the research described in this deliverable has four main *objectives*:

1) Devise a method to inject pedagogical information into a game for learning or ITS based on crowdsourcing teacher expertise.

2) Develop a new algorithm to infer human-readable models from rank reports which builds expressive (white-box) computational models.

3) Provide an insight into the lesson planning practices of teachers of students with dyslexia.

4) Create a pedagogical component for iLearnRW that in combination with other modules adapts the game sessions to provide a more effective learning experience.

The remaining of this document surveys related work on adaptation in ITS and games (Section 2), and provides the details of the adaptation mechanism (Section 3), the crowd-sourcing survey (Section 4) and the machine learning algorithm proposed (Section 5). Due to a moderate initial participation of teachers on the survey, we evaluate the models in the currently available dataset (Section 5) but we also keep running the data collection process at the time of writing of this deliverable. Hence, the final models to be included in the iLearnRW system in month 24 of the project (September 2014) will be described in an updated version of this deliverable submitted before the integration of the system (during summer 2014). This, and other future activities — including publication plans for this work — are described in Section 6.

2. Related Work

We analyse the research related to this deliverable from two separate perspectives. First, we locate our work within research on adaptation in learning technologies. Second, we highlight the contributions to the machine learning community with regards to the method proposed.

2.1 Adaptation in learning technologies

Adaptation has been for many years the main research objective in intelligent tutoring systems following the idea that an effective learning experience should be tailored to the student. As mentioned in the introduction, four modules can be often distinguished in an ITS and different forms of adaptation can be achieved in each of them.

While research on the *student* and *domain expert* modules is vast (e.g. (Baker, 2007; D'Mello, Lehman & Graesser, 2011; Conati and Zhou, 2002; Grafsgaard et al., 2013)), we will not review that work here since this deliverable is not concerned with these parts of the system. For more details about that work can be found on the User modelling deliverable (D4.1)

Regarding the *communication* module, significant efforts have been put towards creating more effective virtual tutors that express affective states (e.g. (Baker et al., 2006; Robison, McQuiggan and Lester, 2009)) or maintain dialogs that resemble human to human interactions (e.g. (Evens et al., 1997; Graesser, 2004)). Our game does not feature tutors and instead, learning is supported by presenting games that require students to perform literacy-related exercises. Nevertheless, iLearnRW provides basic tools (yet fundamental) of text presentation customisation (see Content presentation and adaptation deliverable for more details, D4.2).

The focus of this deliverable is on the *pedagogical* module which is responsible for planning the lessons (relying on the *student* and *expert domain* modules). This module is particularly important for content sequencing tutors where the curriculum needs to be taught along several sessions. In some ITS, this module is not present and the system relies on the teacher to pace the progress (Vanlehn, 2005). We are avoiding this approach in order to provide systems that can provide one-to-one, personalised, teaching without the need of a human teacher at every step of the way.

The simplest solution that does not involve constant teacher supervision was given in ELM-ART (Brusilovsky, Schwarz & Weber, 1996) and ELM-PE (Weber & Mollenberg 1994), two tutoring systems to teach programming; they feature a simple adaptation of the interface where the students can access any content (texts and exercises) but colours identify the appropriateness of contents according to a student model. In this project we aim to design an automatic method that does not require the student to choose the literacy content of her activities.

Regarding autonomous adaptation mechanisms, it is worth mentioning that the lesson planning task has been divided in some studies into *micro* and *macro* adaptation. Macro-adaptation (Shute, 1993) refers to the selection of the general teaching strategy (e.g. teaching through examples or through theory) while micro-adaptation takes care of the low level decisions (e.g. selecting the next curriculum

element). The macro-adaptation layer is not present in our system as the game only supports teaching through exercises.

Macro-adaptation is not that common, and most ITSs rely on a *problem-test-feedback* cycle (Shute, 1995) that is combined with micro-adaptation based on a simple progression algorithm (see Anderson, 1995; Koedinger, 1997; Aleven et al. 2004 among others). In short, the system presents the theory related to the current curriculum element (problem), then the student solves an activity about this element (test) and then the system informs the student about her performance (*feedback*). If the element has been learned successfully, then the system moves on to the next curriculum element, otherwise it repeats the same element again. In this project we search for a more advanced mechanism that rather than only focusing on progression, it counterbalances it with motivation and overlearning.

In *Stat Lady* (Shute, 1995), an ITS to teach Math, an addition to the simple progression algorithm is devised in order to speed up the transition through elements mastered outside the system and to help students who get stuck in a particular exercise. Rather than relying on handcrafted rules, we use a crowdsourcing survey to create a similar module that will decide how to alternate materials over the overall linear curriculum.

In Baker et al. (2006), supplementary exercises on the current topic are injected if the system detects that the student manages to find the correct solution to a problem through gaming (e.g. the student tries all the possible solutions really fast). In our system, gaming/cheating is not an issue as additional motivation to get the correct answer quickly is encouraged through game mechanics and rewards. Furthermore, the complete lesson plan is created *on the fly* opposed to only injecting exercises in exceptional cases.

Overall – while this is not a fully inclusive survey of ITSs and game-based learning systems – it is apparent that research on the pedagogical component has not attracted much interest within the ITSs and game-based learning in recent years. Within this project we believe that this component is of the upmost importance in order to enable games and learning applications to handle large curricula effectively. This deliverable focuses on this module, and introduces a general method to create it automatically based on an expert survey.

2.2. Decision trees and preference learning

Machine learning (Mitchell, 1997) (ML) is a branch of artificial intelligence concerned with computational methods for *automatic learning* from data. These methods can be applied to find unknown relations among the variables or features in a given dataset; in particular, *supervised learning* algorithms can learn a function or model from a set of training examples that associate a set of input features with a set of target outputs. The learning process synthesises the mapping between the inputs and the output.

Depending on the nature of the output, supervised learning algorithms can be classified as *metric regression* if the output is a continuous value, *classification* if the output is an item from a finite set (class) and *preference learning* if the output is an ordered set, ordinal class (rank) or ordinal relation.

In this deliverable, we deal with ranks annotated by teachers, which restricts our tools to preference learning algorithms (Fürnkranz & Hüllermeier, 2010). These include powerful learning techniques

such as Gaussian processes (Chu & Ghahramani,2005; Nielsen, Jensen, & Larsen, 2011; Abbasnejad, Bonilla & Sanner, 2011), support vector machines (Joachims, 2002; Radlinski & Joachims, 2005; Bahamonde et al., 2007) and artificial neural networks (Pedersen, Togelius & Yannakakis, 2009). While these methods provide state-of-the-art prediction accuracies, they lack on *expressivity*; i.e. it is not trivial to interpret the relations between inputs and outputs captured by the model.

In that respect, decision trees (Quinlan, 1986) are likely to be the most expressive learning algorithm as they generate a tree structure that can be interpreted effortlessly as a set of human-readable rules. Decision trees have been extensively used as a data modelling tool (Rokach, 2008) but they can only be trained using classes or continuous values. In this deliverable we introduce a new preference learning algorithm that allows us to construct decision trees based on ordinal data. This both helps us to create transparent adaptation mechanisms and – beyond the domain we investigate here – adds scientific value to preference (machine) learning at large.

3. Overview of the adaptation mechanisms

One of the most innovative features of the iLearnRW system — due to its lack of existence in other similar teaching tools for dyslexics — is its ability to recommend and plan the specific difficulties and activities a specific student should work on to improve her individual educational needs. When playing the game, the student will meet characters that casually request her help to solve particular problems or participate in events occurring in the virtual world. Note that these events and character encounters are not hardcoded and in fact, they will be prompted following a personalised teaching plan. This plan is assembled through 3 (conceptually) separated layers: a **curriculum** which details the order in which specific difficulties are more easily mastered, a **user profile** that tracks the progress of the student over the curriculum, and a short-term **lesson planner** that provides an optimal learning experience by varying the activities, specific difficulties and level of challenge in accordance to the user model, the curriculum and the recent performance of the student in the game. Ultimately, a **content presentation** module delivers the content through an adequate and personalised interface. The relation among these components is depicted in Figure 1.

3.1. Curriculum (expert knowledge component)

During the first year of the project a set of specific difficulties that Greek-speaking and English-speaking learners with dyslexia face were identified. These are organised in language areas, and within each of them, the specific difficulties are ordered according to complexity (and thus, the most logical teaching order).

To facilitate the construction of the lesson planner, the language areas are flattened yielding a single list of specific difficulties for each language. The experts at EPIRUS pointed out that for Greek the language areas are themselves ordered, which leads to a trivial flattening (the list of specific difficulties in one language area follows the list of difficulties of the previous area). The combination is not as simple in English according to the collaborators from Dyslexia Action, yet it follows the teaching points that were already specified in D4.1. While the organisation by learning areas leads to a more intuitive user profile, flattening them is a natural procedure from the lesson planner perspective as it better represents the linear progression of teaching in the classroom.

3.2. User profile (student model component)

The user profile contains a level of severity for each of the specific difficulties: mastered difficulty (level 0), difficulty that needs reinforcement (level 1), difficulty that has been presented but is not overcome (level 2) and difficulty that has not been presented or the student has made poor progress (level 3). Based on this information, the user profile provides the current teaching point for each language area (i.e. the first difficulty with level of severity equal to 3).

In D4.1, rules for downgrading the severity of a difficulty were proposed. In addition to those, upgrading rules will be incorporated to support the lesson planner functionalities. In particular, two rules have been proposed in line with current teaching methods: (1) the severity associated with a difficulty will increase if the student achieves a poor score on a related activity and, (2) when a

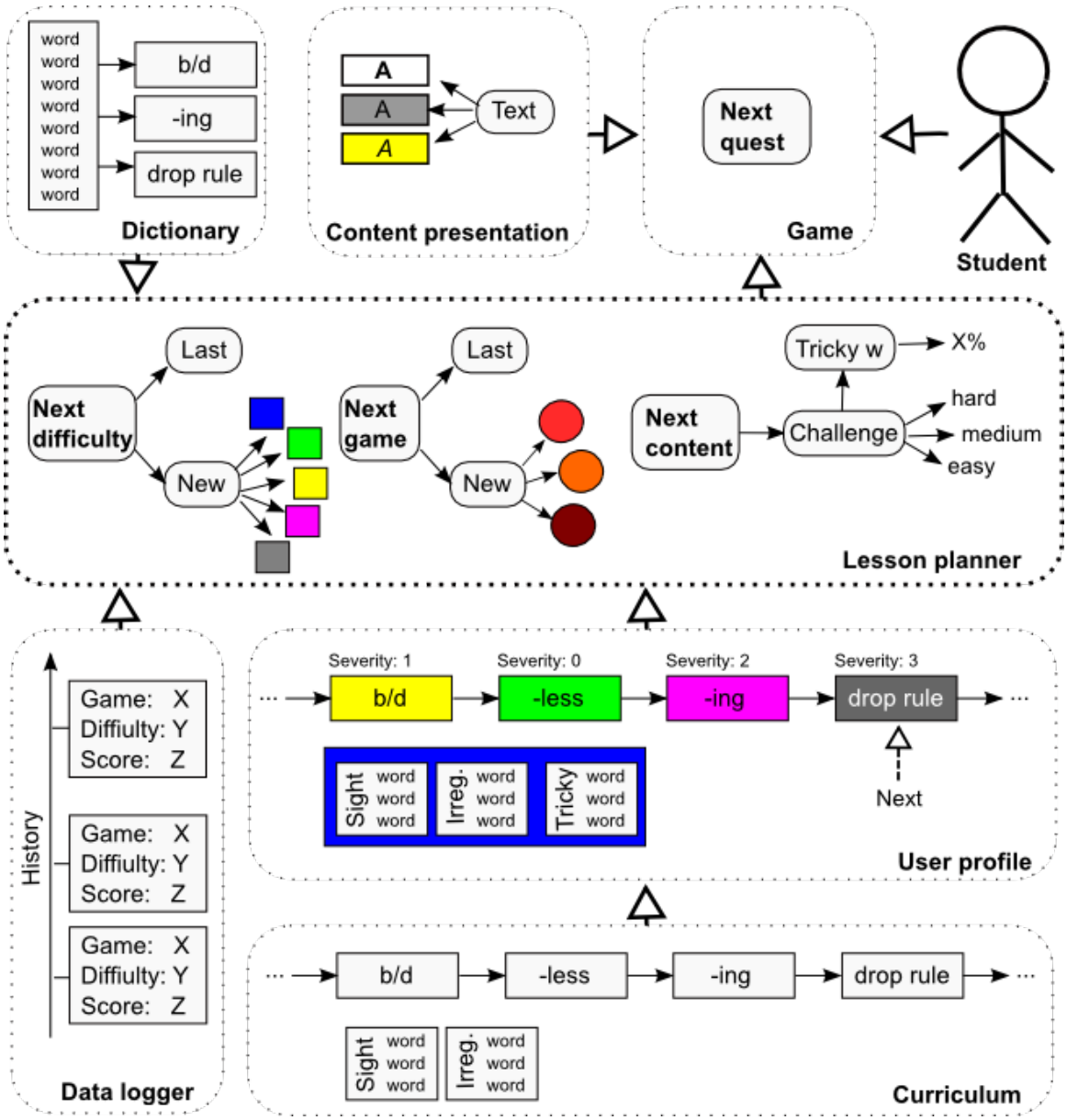


Figure 1. Components that contribute to the adaptation in the game. This deliverable focuses on the lesson planner which selects the most appropriate activities and difficulties based on the previous performance of the student (stored in the data logger) and the level of severity of the specific difficulties (provided by the user profile).

difficulty associated with a level of severity equal to 0 has not been presented for a long period, the severity is upgraded to level 1.

An additional element of the user profile relevant for the lesson planner is the list of words that are problematic for the each individual student (tricky words); as this list is not assigned to a particular teaching point in the curriculum, we regard it as a special difficulty (referred to as *tricky words*) for the consideration of the lesson planner.

3.3. Lesson planner (pedagogical component)

The adaptation of the game scenarios is directed by the lesson planner. This module starts to function once the student logs into the game and it constantly estimates which learning activity should be recommended next in the *Ghostbook* (i.e. the play mode) or injected into the story in the game world (i.e. adventure mode).

In particular, this module will adapt the game through the following 3 functionalities:

1) Automatic selection of the next specific difficulty that the student should work on.

2) Automatic selection of the game activity and level of challenge that should be used to work on that difficulty.

3) Automatic selection of the words or sentences to appear in the activity.

The selection of the next difficulty in this component is not based on the exact difficulties found in the curriculum, but based on their level of severity. A computational model is used to predict the severity of the difficulty that should be presented next; in more detail, the output of the model indicates whether the student should work on (a) the same difficulty as in the last game, (b) an difficulty not presented before, (c) a difficulty already mastered, (d) a difficulty that requires reinforcement or (e) the tricky words difficulty. This information is then combined with the user profile which produces a set of candidate difficulties from which the game can choose randomly. For instance, if the model estimates that the student should work on a difficulty already mastered, then the game can choose from any of the difficulties associated with severity 0 in the user model.

Once the specific difficulty is selected, the planner proceeds to select a game activity. For each specific difficulty a number of games are enabled, and the planner utilises a predictive model to estimate the most appropriate and its level of challenge. In particular, the output of the model indicates whether the student should play (a) the same game as last or (b) a different game, and its challenge level, i) easy, ii) medium or iii) hard. The level of challenge in the games can be affected by the complexity of its literacy content but also by adjusting particular parameters of the mechanics (e.g. the amount of time that words are visible in whackamole). As the output is left generic, the model can potentially be combined with any type of activity after the designer or teacher have identified different levels of challenge. The different levels of challenge for the initial version of the activities were introduced in D3.5 but the Serious games deliverable (D5.3) will contain the levels of challenge for the final game scenarios.

With the difficulty and the game fixed, the final choice is the content to play with. The services included in the server provide a list of words or sentences ordered by complexity for each specific difficulty. Thus, the selection of specific difficulty and level of challenge would be sufficient to query the server and pull appropriate content to fill the activity. However, we add an additional model that estimates the amount of tricky words to be included in the activity; in particular, from the total number of words used in the activity, whether (a) none, (b) less than half, (c) more than half or (d) all of them should be selected from the tricky words list. Furthermore, for activities where distractors are required (i.e. words from other difficulties that are mixed with the main difficulty presented), a final model estimates the severity of the secondary content, making a prediction among the same options as the selection of difficulty.

The predictive models utilise information from the activities played previously such as the severity of the specific difficulties, the game and challenge level, the amount of tricky words used and the severity associated with distractors. In addition to this contextual metrics, the student performance on those activities is also fed into the model. Following the suggestion from the dyslexia experts in the consortium, the performance is divided (when possible) into accuracy and speed.

Additional inputs – such as physiological signals or verbal cues – providing explicit information about the mental state of the student could have been beneficial, and in fact, they were contemplated in the early stages of the project (Martínez, Bengio & Yannakakis, 2013; Knight, Martínez & Yannakakis, 2013). However, this idea was dropped in favour of a system that can be easily distributed to schools around the globe as it is not limited by hardware beyond a tablet (e.g. physiological sensors). Within the tablet, one could use as input the camera to detect affect, but given the additional privacy issues that arise from such addition, together with the restrictions that it imposes to users (i.e. play in well-lighted rooms well-in-front of the tablet camera) we decided to focus our research efforts on a different front. Finally, as the game does not make use of speech as input, affect verbal cues are directly not available.

In sum, the pedagogical component is composed by a set of predictive models that select the optimal elements for the next game using information about the previous game sessions, and combining information from the user profile and the server services.

3.4. Content presentation (communication component)

Three main features are integrated within this final component. The first one is related to the teaching strategies which detail how feedback and tutorials need to be provided and emphasises the importance of the multi-sensory approach to learning. The second one refers to general knowledge about content presentation for readers with dyslexia including for instance using short sentences and particular vocabulary. While these features are of great importance, they do not require tailoring to each student and will be simply integrated in the game development process.

The third feature relates to the parameters for presenting text on the screen, such as font, background and speed. Evaluating automatically the preferred parameters for each student within the game is a highly complicated task that might go in detriment of enjoyment (i.e. student might need to go through a tedious calibration process involving seemingly unreadable text). Instead, we decided to provide manual customisable options which are commonly used in software accessible for readers with dyslexia (see Content adaptation and presentation deliverable, D4.2).

4. Crowd-sourcing a lesson planner

In line with the other design activities within iLearnRW, we have also involved users in the creation of the predictive models that support the lesson planner. In particular, we devised an on-line crowd-sourcing survey that has been distributed to teachers with experience on one-to-one dyslexia interventions in order to capture their practical wisdom.

In this survey, experts are not directly asked about their strategies with regards to overlearning or motivation. Instead, the survey presents case scenarios in which the teacher has to propose activities and difficulties for a hypothetical student whose speed and accuracy for each game are known. We believe that this format is advantageous because the knowledge of teachers about overlearning and motivations is expected to be intrinsic to their practise as opposed to self-aware methods. In addition, the information obtained with the scenario-based questionnaire is given in terms of a lesson plan built around games; hence we can train the model for our system without additional human expertise to interpret the results or translate them into a usable format.

As the scenarios provide only information about the performance of the hypothetical student, the teachers are forced to implement a lesson plan disregarding physical manifestations of affect from the students. As affect sensing is out of the scope of this project, this provides more valuable information to the models. On the other hand, it requires teachers to imagine that situation and put themselves into the right context; we believe, however, that despite this mental exercise, the implicit knowledge that teachers have developed throughout years of experience will be captured with this survey. The survey is formed in such a way to enable a context-rich situation for teachers and assist them relating to the situation at hand.

In the remaining of this section we highlight the main features of this experiment including the protocol followed (section 4.1), the questions asked (section 4.2), the format of answers provided (section 4.3) and details about the participants of the crowdsourcing experiment.

4.1. Protocol

The survey starts with a simple consent form that the participant needs to accept before proceeding to the questionnaire (see Appendix A.1). After few pages describing the format of the survey (see Appendices A.2 through A.7), the participant is asked to provide basic demographic information, i.e. age, gender, number of years in education and number of years in dyslexia education (see Appendix A. 8). Optionally, the participant can provide her email address in order to receive further updates on this study.

Once the demographics are collected, the participant is presented with 5 consecutive teaching scenarios (see Figure 2). For each of these scenarios, the participant has to select a difficulty, a game, and its content; after that information is complete for one scenario, the participant is presented with the accuracy and speed obtained by the hypothetical student and asked to prepare the next scenario based on that information. The performance of the hypothetical student is selected by sampling (using a uniform random distribution) all possible combinations of speed (high and low) and performance

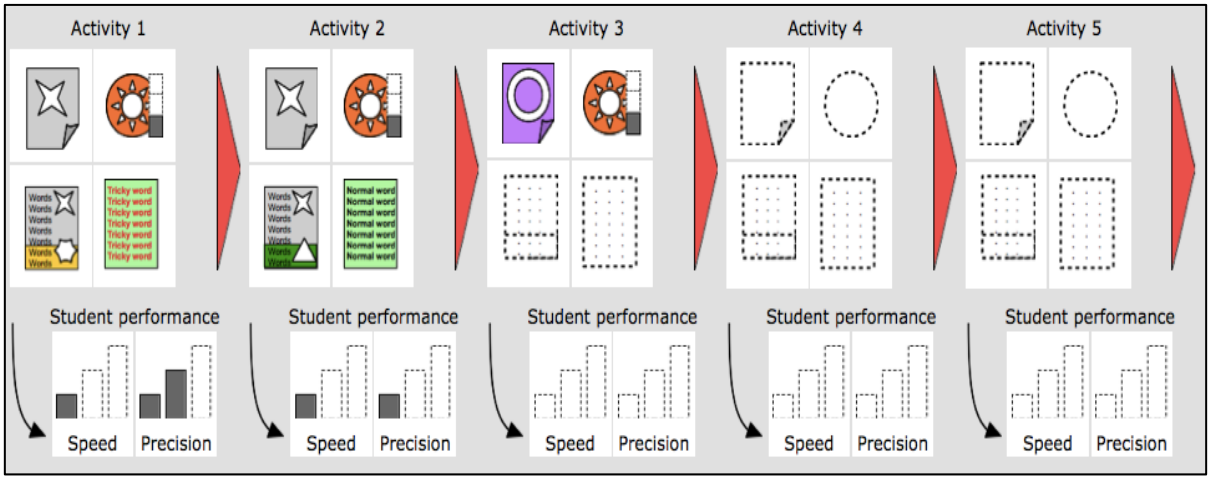


Figure 2. Example of the progression view displayed during the questionnaire. For each of the five scenarios, the participant is asked to answer four questions and her choices are represented with particular icons. Once the variables of one scenario/activity are completed, the questionnaire generates the speed and precision of the hypothetical student and questions about the next activity are queried.

(high, medium, low) across 4 consecutive scenarios (the performance of the last scenario is irrelevant) and taking into account the participant choices. Obtaining data for all possible combinations is unrealistic, but by sampling uniformly the space of possibilities, we can create a dataset that is representative of the models we try to infer resulting to more general models.

After the participant has completed the 5 scenarios, a final page thanks her for participating and offers the possibility to replay a new sequence of scenarios.

4.2. Questions content

The questions presented for each scenario are formulated at the same semantic level used in a lesson planner – i.e. participants are not asked to choose one difficulty over the 100 difficulties existent in the English profile, or to choose the exact words that will be used in the activity. Note that, in practice, teachers rely on structured programs such as DILP to prepare their classes, and therefore the questionnaires provided are at the right abstraction level. While teachers might normally operate with more contextual information about the student, the concepts included in the questionnaire are basic for reacting to the student progress during a teaching session, thus it should be easy for the participants to relate to the scenario. Inevitably, inconsistent answers or noisy reports may be introduced in our data sets through this process. An efficient machine learning method, however, is able to identify patterns that are significant across multiple teachers (and thereby more data points) in the survey with the assumption that the crowdsourced data obtained is representative of the function we attempt to approximate via machine learning.

Regarding specific difficulties, the participant can choose from *seven* options across the 5 scenarios, namely: (1) the furthest difficulty in the curriculum that has been presented to the student (related to severity 2); (2)(3) two different difficulties that need reinforcement (severity 1); (4)(5) two different

difficulties that have been mastered (severity 0); (6) one difficulty that has not been taught before (severity 3); and (7) frequent and tricky words. We offer two different options with severity 1 and two options with severity 0 because we anticipate that these difficulties will be often selected (as part of overlearning). By providing two separate options for each level of severity, we allow participants to create lessons with certain level of variation even if the class is only focused on overlearning. The frequent and tricky words category is included as it is not within the ordered curriculum and thus, it would not be accessible from a pure severity perspective. An example of this question is depicted in Appendix A.9 (left side).

The question regarding the game is also generic: rather than choosing among specific games, the participant can choose between two games and three levels of challenge for each of them. One of the games is presented as preferred over the other by the student in order to give to the participant meaningful information about the games without mentioning details about the mechanics (see Appendix A.9, right side).

With regards to content, the participant needs to specify the amount of words in the game that should be selected from the tricky words list (options are none, less than half, more than half and all). In addition, the participant has to choose the severity of the difficulties used as distractors; the options are the same as when selecting the specific difficulty. These two questions are depicted in Appendix A.10.

4.3. Answer format

For each question, the participant has to drag and drop the different option into a ranking (see Appendix A.7 for two examples). Alternative to only selecting the *preferred* option, the ranking format allows us to obtain much more information beyond just *what the best option is*, including for instance what is the worst option and whether there is no difference among some options (i.e. several options might present the same ranking). Compared to rating-based questionnaires, rankings provide less unbiased information as they do not require the participant to quantify their preference into an absolute scale (Yannakakis & Hallam, 2011; Ovadia, 2004). Note that that quantification process is affected, among other factors, by interpersonal differences, which interferes with any attempt to create a model across several users. On the other hand, the greatest disadvantage of rankings is usually their higher cognitive demand required by the survey participant; to minimise this, we allow participants to only rank a subset of the options. This does not only reduce the cognitive effort required to complete the questionnaire, but it also helps us minimising the number of unclear preferences.

4.4. Participants

At the moment of writing, the survey is still undergoing in an attempt to maximise the number of possible participants before the final models are deployed. To date, the questionnaire has been distributed among teachers working with Dyslexia Action and EPIRUS, and teachers from the Specific Learning Difficulties Service unit (SpLD), the group that provides dyslexia diagnostic tests and education support all over Malta. So far the number of teachers that have completed the survey is 5 and the number of ranks obtained is 189.

5. Preference Learning Decision Trees

In order to find regularities in the information collected through the survey and generalise it to a set of computational predictive models, we decided to use a machine learning methodology based on **decision trees** (Quinlan, 1986).

A decision tree (DT) is a basic computational model typically used for classification tasks. Unlike other methods such as artificial neural networks, the output of the decision tree is not directly connected to the exact value of the input features (e.g. the accuracy of the student in the previous played activity). Instead, the output is connected to particular intervals of values for each feature. Consequently, decision trees cannot differentiate among input samples with the same level of granularity as other methods, which it may turn in lower prediction accuracies on particular domains. On the other hand, DTs offer an excellent level of **model expressivity** that enables humans to directly interpret the models obtained. This interpretation phase is key for understanding and validating the lesson plan module derived from the opinions of different experts. In this context, we expect that the learned DTs will draw the mapping from the information extracted from previously played activities into an estimation of the quality of different possibilities for the next activity.

DTs are used for regression and classification tasks but in this work we rely on rankings. This data format eliminates the possibility of using the standard training algorithms. One could transform the ranking problem into a classification task (i.e. each rank is a separate class) or regression task (i.e. each rank is a numerical value), but this approach adds a number of experimental biases (Martínez, Yannakakis & Hallam, 2013). Instead, in this section we propose an adaptation of standard decision trees that learn their structure from partial orders such as pairwise preferences and ranks.

5.1. The model

A standard DT contains a number of non-terminal nodes, each of them connected to one feature. The outgoing branches of the node define a complete split of the possible of values of the associated feature. Input samples are sorted down the tree by following the feature-value conditions specified by each branch. Terminal nodes or leafs are assigned to a single class (in classification tasks) or real-value (in regression tasks) which are assigned to the samples that fall within that node.

The structure of the tree is practically the same for rankings, with the only difference that leaf nodes define ranks (i.e. ordinal values) instead of classes or real numbers. Figure 3 depicts an example of decision tree for the task of selecting the severity of difficulty in the next activity. In this example, the severity of the difficulty presented on the previous activity is analysed in the root node. If the severity was 2 or 3, then the decision of the next severity depends on the accuracy whilst if the severity was below 2, the decision depends on the speed. The training algorithm automatically finds the features that are more relevant for the decision in each case; this example tree suggests that if the student was working on a difficulty already mastered or that needs reinforcement, the reading speed is sufficient to qualify the next severity, whereas if the student was working on difficulties with higher severities, then the accuracy is more important to make the decision.

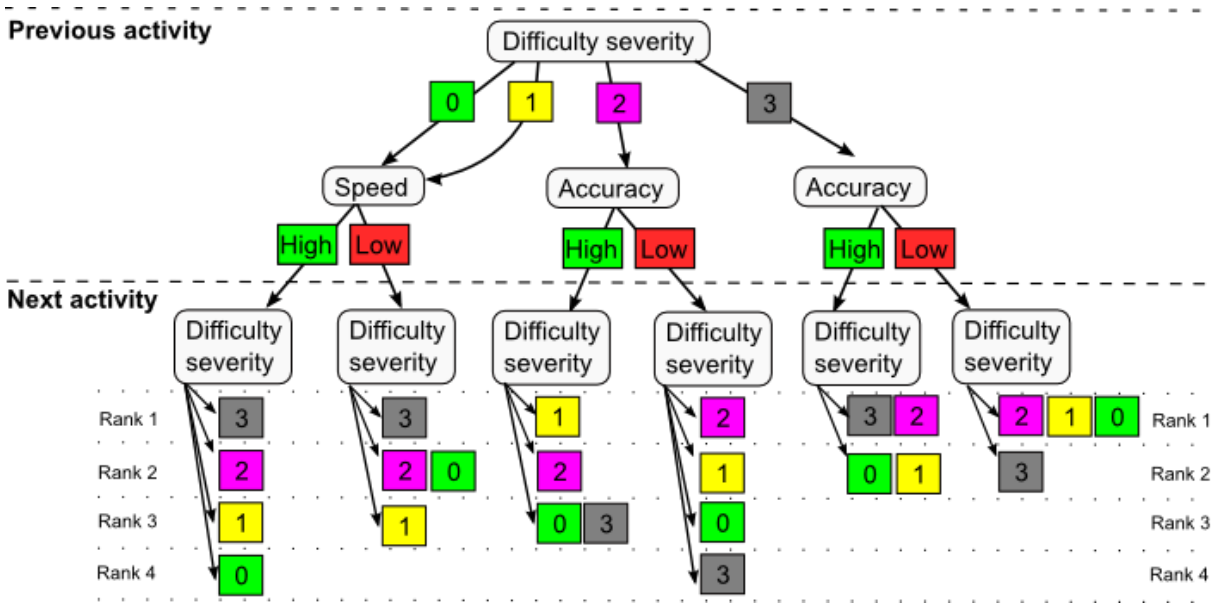


Figure 3. Example of a decision tree that predicts the optimal (rank 1) severity for the next activity based on the speed, accuracy and severity in the previous activity.

The terminal nodes in the sample tree are ranks from 1 to 4, and they are connected to each of the values branching out from the last non-terminal nodes (in this example, the severity that the game should present for the next activity). Note that the ranking does not indicate in which order the severities will be played in subsequent activities, instead the ranking quantifies the quality of the different options when they are being considered as the next severity. Thus, for the selection of the next severity we only need those branches that are labeled as rank 1. For instance, when the previous severity was 2 and the student obtained a low accuracy, the next activity should feature a severity of 2 again.

The other ranks in the tree, while are not directly applicable to the lesson planner, they can provide interesting insights on the experts' practices, such as what would be the worst choice in each case.

All the information that was made available to participants in the crowd-sourcing survey is a potential input for the DT, i.e. the speed and accuracy of the student in the previous games together with the composition of those games (severities, level of challenge, game and words used). The 5-scenario length used in the questionnaire allows us to create different models with variant level of memory, i.e. models can make the prediction based on the previous game only or any number of previous games up to 4. A comparison of the cross-validation accuracy of these different variants will reveal the amount of past information that the teachers used to make their decisions during the survey.

5.2. The training algorithm

Standard methods to train DTs such as ID3 and C4.5 rely on metrics dependent on the prediction accuracy on a particular node. The metric indicates whether it is beneficial to the prediction accuracy to expand terminal nodes with a feature not considered in existing branches in the tree.

The training algorithm that we propose for ranks follows the same principle as ID3 and C4.5 but relies on metrics that measure the number of correctly classified **pairs**. Note that a ranking can be transformed into several pairwise preferences without loss of information.

When a node is expanded, the resulting leaf nodes are sorted in a particular order; thus the feature and ordering of its possible values (or possible value intervals) that yields more correctly classified pairs is selected. The expansion of nodes follows a depth-first strategy, always starting from the nodes ordered last. In this way, node expansions cannot alter pairs already classified by previous levels; instead, it only classifies pairs of input samples that fall in the same node (and thus no preference among them had been yet defined).

Consequently, pairs that are classified incorrectly early in the process cannot be corrected. Therefore, the metric used needs to account not only for correctly classified pairs, but also incorrectly classified and not classified. For instance, given two alternative node expansions that yield the same number of correctly classified pairs, the one with a lower number of incorrectly classified (and thus larger number of not classified) should be preferred. For this initial study, we use Kendall's Tau ($\boldsymbol{\tau}$) defined as the difference between concordant and discordant pairs between two orders, divided by the total number of pairs (Kendall, 1938). More formally, we calculate the metric as follows:

$$\boldsymbol{\tau} = (n_{correct} - n_{incorrect})\,(n_{correct} + n_{incorrect} + n_{tie}) \quad \textbf{(Equation 1)}$$

where $n_{correct}$ denotes the number of correctly classified pairs, $n_{incorrect}$ denotes the number of incorrectly classified pairs and n_{tie} denotes the number of pairs with unknown preference. For more details, the algorithm pseudo-code is detailed in **Appendix B**.

5.3. Evaluation

Although we have not yet gathered enough data to build a robust lesson planner, in this section we report an initial evaluation of the algorithm by comparing it to a standard preference learning method. We use backpropagation to train artificial neural networks as this method has yielded state-of-the-art results in other domains (Burges et al., 2005) and will, therefore, serve as an excellent baseline method to compare against. We use the data collected so far to train four independent predictors of the severity, the game, the severity of distractors and the amount of tricky words in the next activity.

Each ranking (i.e. the answer to one question in one of the five scenarios) is converted — without loss of information — to a set of pairwise preferences. For instance, if a participant ranked four options {A, B, C, D} as {rank #2, rank #3, rank #2, rank #1}, then we create the pairs (D, A), (D, B), (D, C), (A, B) and (C, B), where the first option in the pair is *preferred* over the second; note that since A and C are ranked the same, no relation among them is included (as there is no clear preference for one over the other). It is also worth noting also that rankings are relative measures (i.e. rank #3 in the first scenario might have a different meaning than rank #3 in the last scenario). Hence, matching rankings across scenarios and across participants should be avoided. Therefore we only include pairs that involve options from the same ranking. The total number of pairs resulting from this conversion is 91, 139, 91 and 32 for severity, game, distractors and tricky words, respectively.

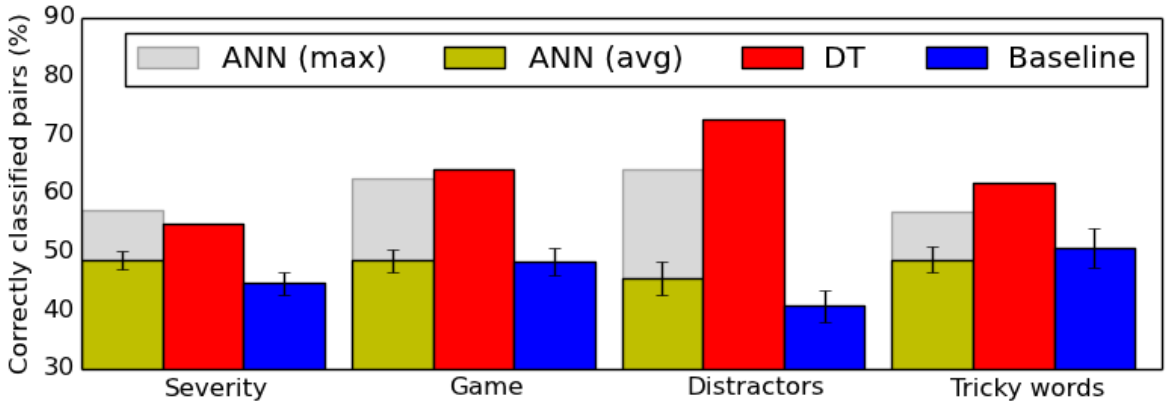


Figure 4. Average 3-fold cross validation accuracy of decision trees (DT) artificial neural networks (ANN) — average and maximum of 10 runs — on predicting the severity, games, distractors and tricky words that should employed in the next activity. Baseline prediction is calculated as the average accuracy of 10 ANNs with random weights. Standard error is displayed as error bars on top of bars representing averages.

We use as inputs the scenario number, the severity and game used, and the precision and speed of the student in the previous activity. For each dataset, we select the best learning rate, topology and number of training epochs for the ANN after an exploratory analysis. The same procedure was used with the DT to select the maximum depth and maximum number of splits for each feature.

Since the training algorithm for DTs is deterministic we report the prediction accuracy of a single DT. On the other hand, the performance of an ANN depends on its initial random weights; therefore, we include in the results the average and maximum prediction accuracy of ten models trained independently. In addition, we provide a baseline performance which is based on chance predictions derived by the average prediction accuracy of 10 ANNs with random weights (i.e. the average accuracy of random functions). The performance measure (i.e. prediction accuracy) used is the percentage of correctly classified pairs and it is evaluated using standard average 3-fold cross-validation.

As depicted in Figure 4, DTs yield models with higher prediction accuracies than the average ANN in all four prediction tasks, and even higher than the maximum in three out of four. While it might be surprising at first that DTs outperform ANNs, this can be partly explained by the feature selection process implicit in DTs; by discarding irrelevant features DTs can achieve a better generalisability. In addition, the performance values of ANNs in this dataset are highly dependent on the initialisation of the weights as the input features for samples in the same pair are always very similar — which leads to deficient training information for the weights connected to them. This characteristic of the data has also an effect on the proposed DT training algorithm, which opens up interesting research directions to enhance the algorithm in the future.

Although the prediction accuracies obtained in this initial evaluation are not surpassing 80%, in all experiments DTs outperformed the baseline prediction suggesting the validity of the method and its supremacy over established preference learning methods in the literature. We expect that with a larger

dataset, the patterns in the data will become more salient, facilitating the creation of more accurate data-driven models. Low accuracies on a large dataset would indicate that there is no agreement among the responses of the teachers surveyed or that in a large number of scenarios the decisions are arbitrary. If that is the case for any of our prediction tasks, we can easily replace the specific machine-learned model with a set of rules hand-crafted by the dyslexia experts in the consortium. Doing so would result to a hybrid adaptation module that contains the very best (generalisable) data-driven models fused with the models created through domain-expertise.

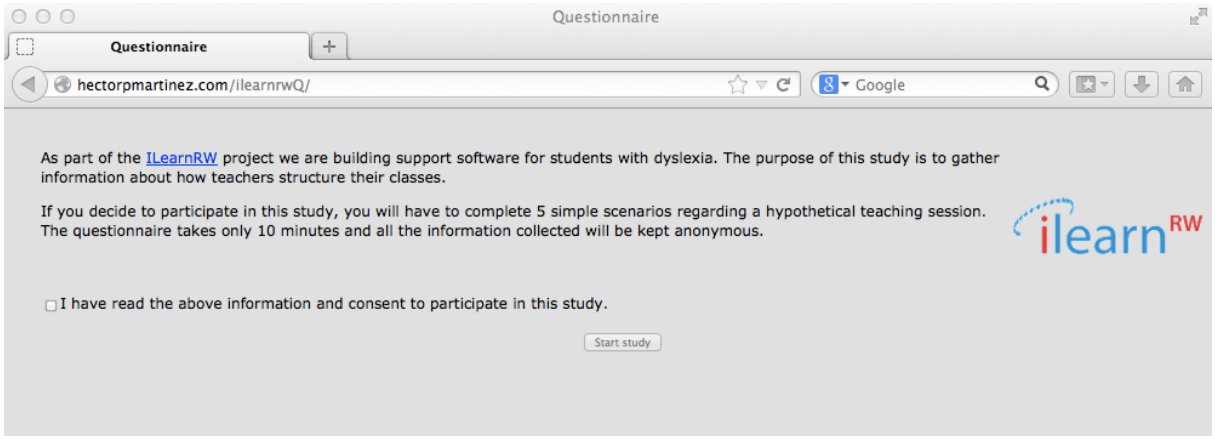
6. Summary and Future Work

In this deliverable we have defined a framework for adaptation of the iLearnRW game. Adaptation affects the specific difficulties and composition of the game activities that are presented at any point in the game by balancing the need for progression through the curriculum with the need for overlearning previously presented difficulties, while maintaining the student motivated and interested. This framework relies on modules defined in other deliverables but requires a new component for lesson planning.

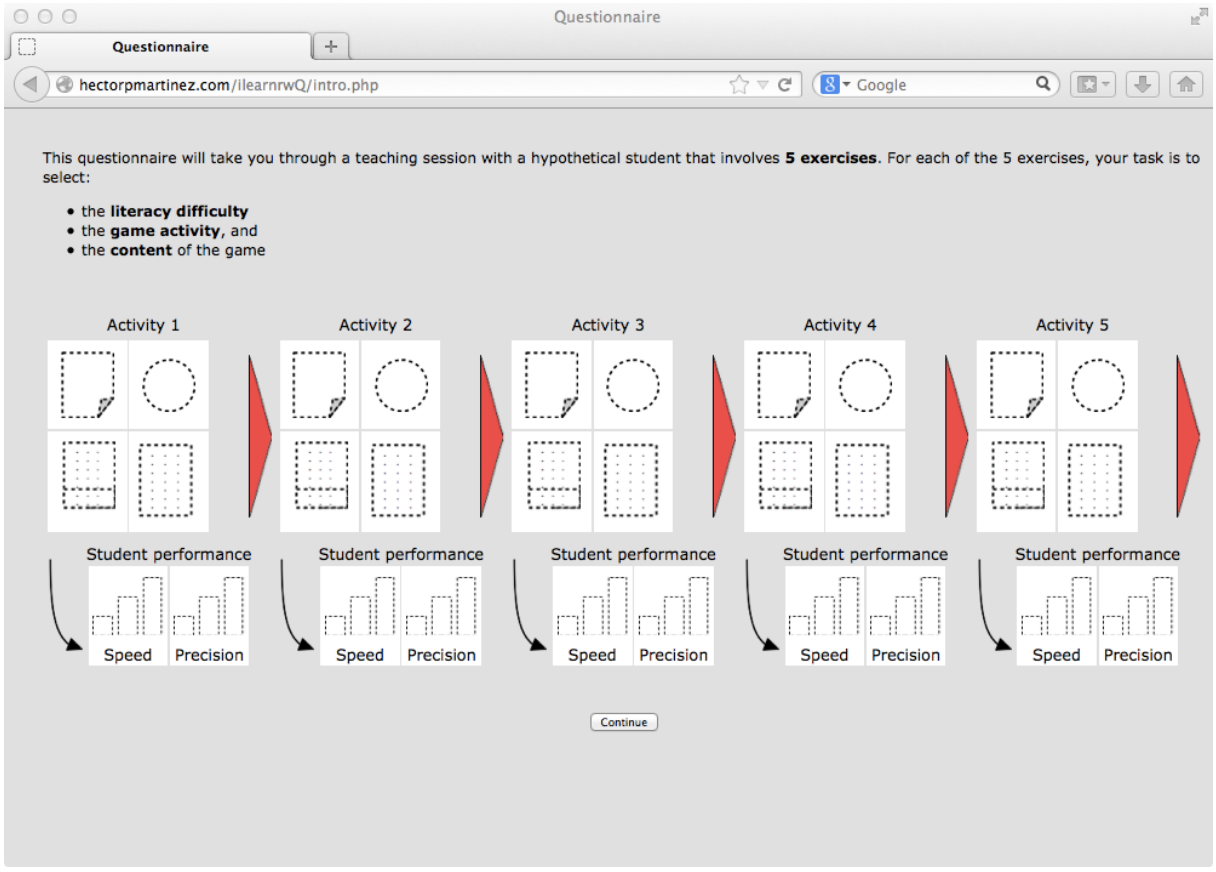
Rather than ad-hoc creating this model based on simplistic notions of progression or using solely the expertise of the partners in the consortium, we propose a data-driven crowd-sourcing survey and developed a new learning algorithm that couples preference learning and decision trees to automatically derive the lesson planner from data. The performance of the decision tree preference learning algorithm on the current dataset suggests that it is both efficient and superior to other benchmark preference learning algorithms (i.e. artificial neural networks). Since (at the moment of writing) only a fraction of the teachers have participated in the survey, the final models that will be deployed with the game have not been created yet. Hence, the obvious next steps of this research consist of training and analysing of the decision trees based on the complete pool of answers from the experts. New invites for the survey are being sent and we are expecting to gather data from up to 25 different teachers before June 2014. The analysis of that data and the final models that will be integrated in the final prototype of the iLearnRW system will be reported in an updated version of this deliverable before its final integration to the system (i.e. before September 2014).

Appendix A: Survey

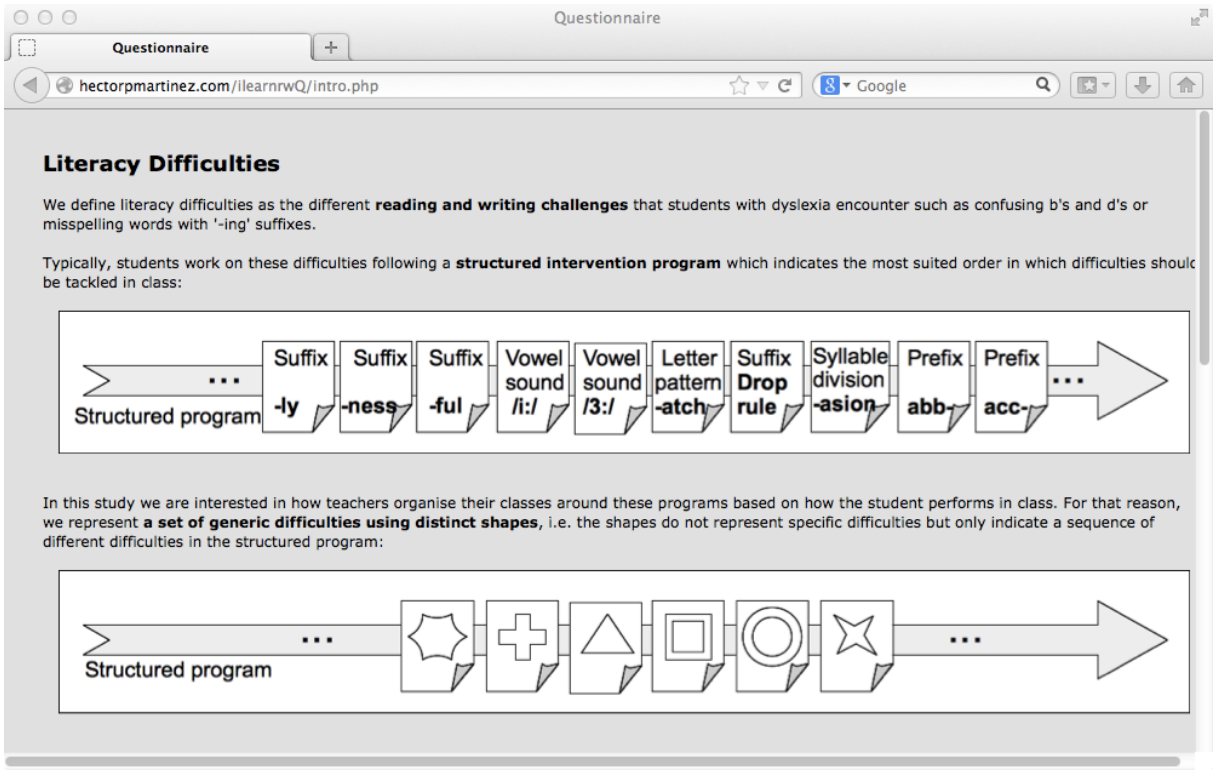
A.1. Initial Screen

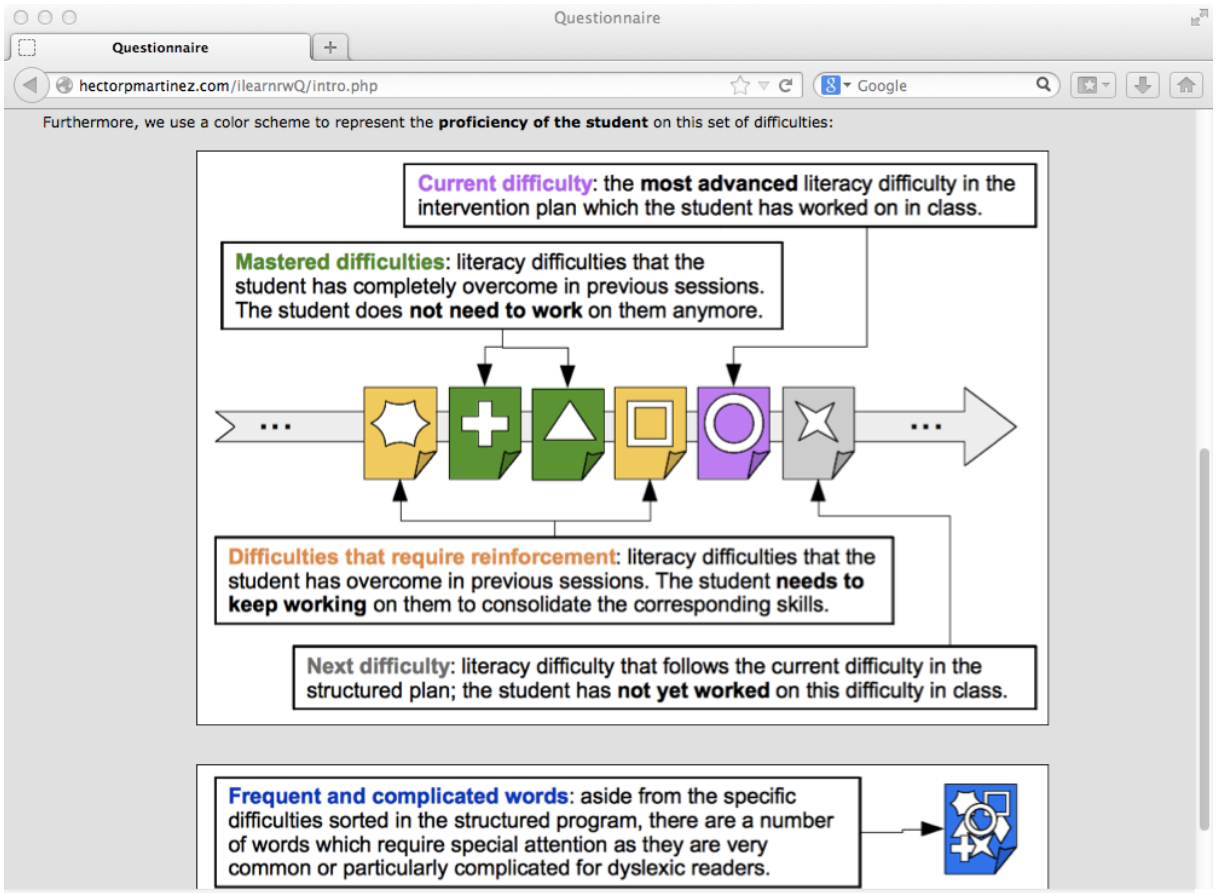


A.2. Basic instructions

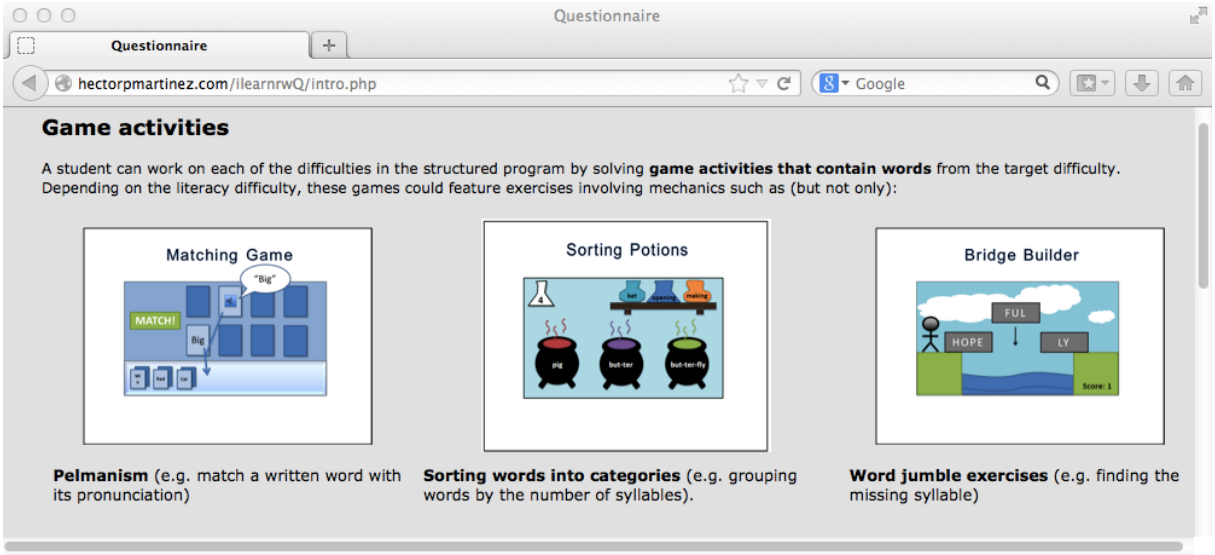


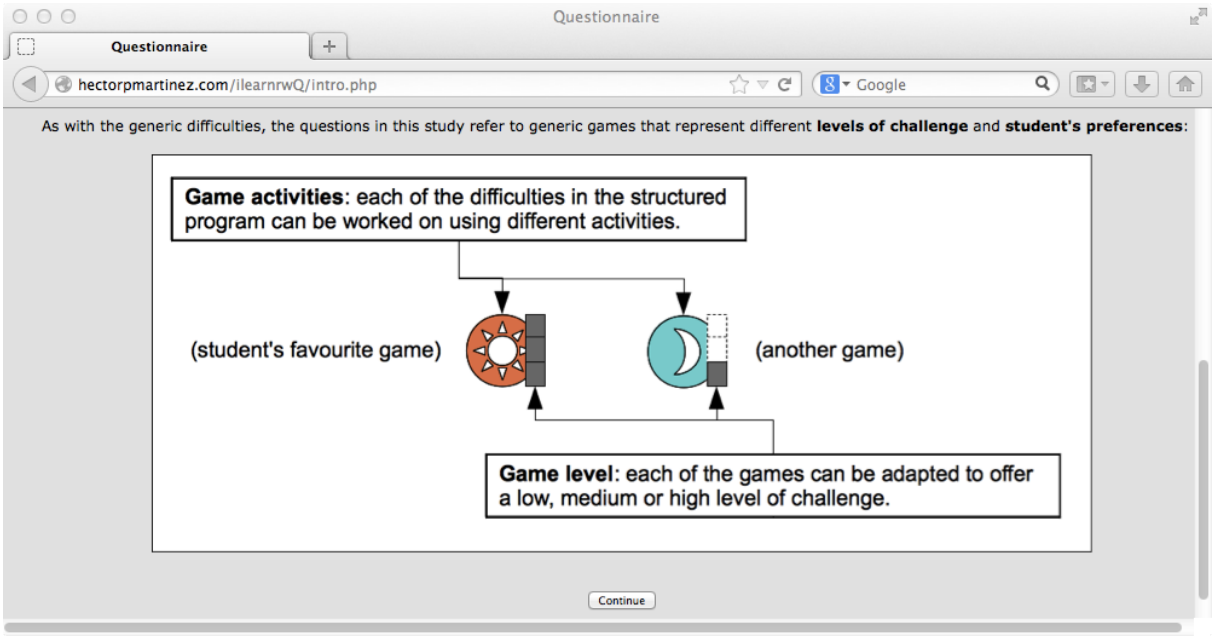
A.3. Difficulties



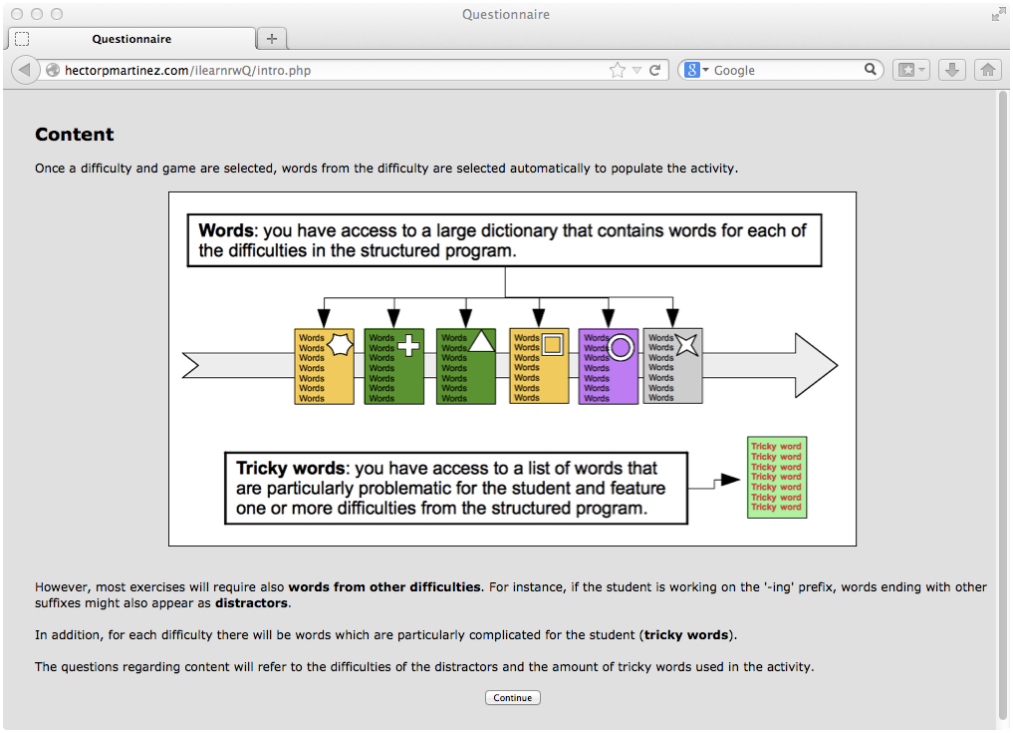


A.4. Games

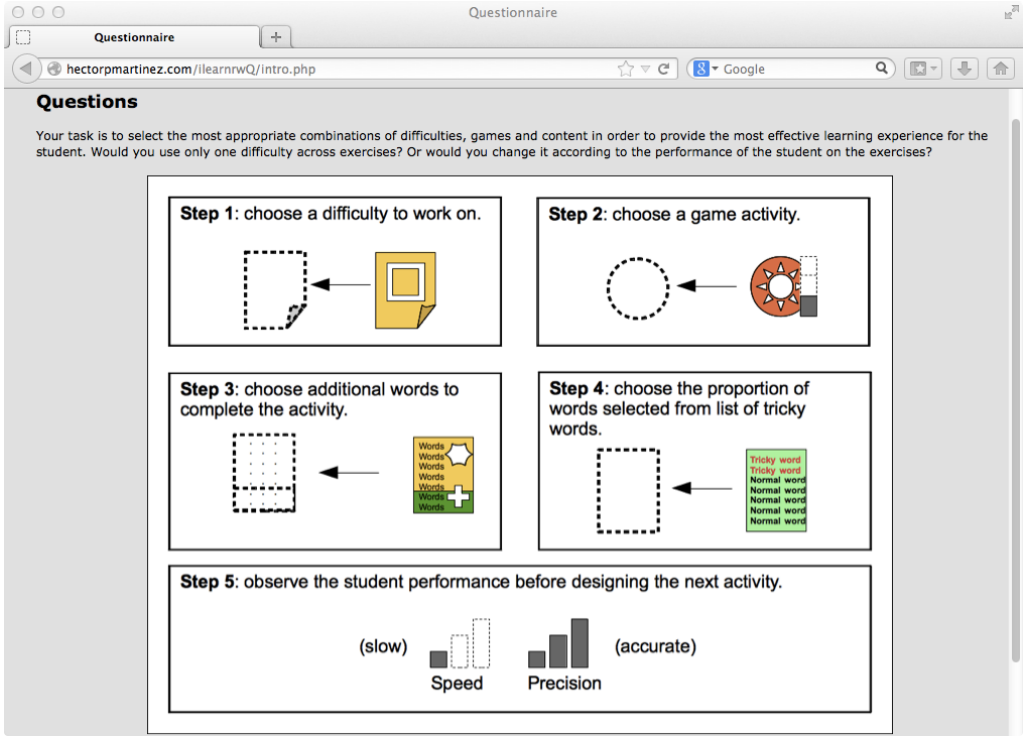

Questionnaire
Questionnaire
hectorpmartinez.com/ilearnrwQ/intro.php
Google
Game activities
A student can work on each of the difficulties in the structured program by solving game activities that contain words from the target difficulty. Depending on the literacy difficulty, these games could feature exercises involving mechanics such as (but not only):
Matching Game
MATCH!
Sorting Potions
Bridge Builder
HOPE
FUL
LY
Pelmanism (e.g. match a written word with its pronunciation)
Sorting words into categories (e.g. grouping words by the number of syllables).
Word jumble exercises (e.g. finding the missing syllable)


Questionnaire
Questionnaire
hectorpmartinez.com/ilearnrwQ/intro.php
Google
As with the generic difficulties, the questions in this study refer to generic games that represent different levels of challenge and student's preferences:
Game activities: each of the difficulties in the structured program can be worked on using different activities.
(student's favourite game)
(another game)
Game level: each of the games can be adapted to offer a low, medium or high level of challenge.
Continue

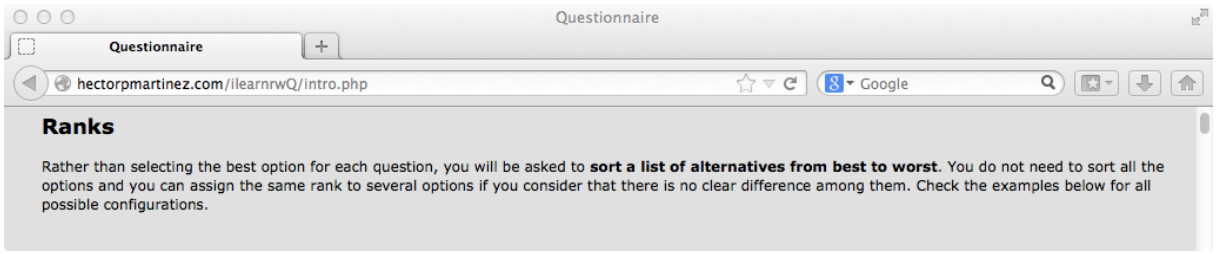
A.5. Content

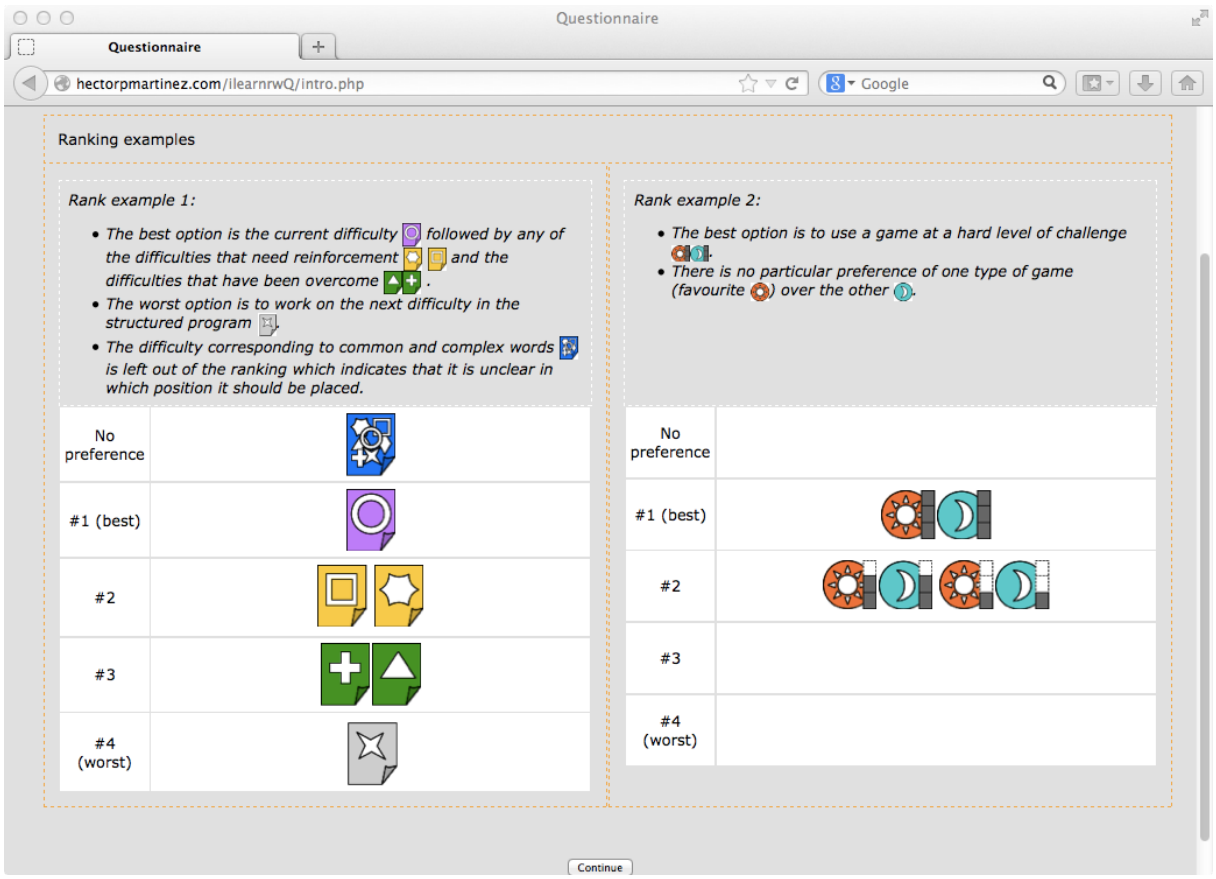


A.6. Question format

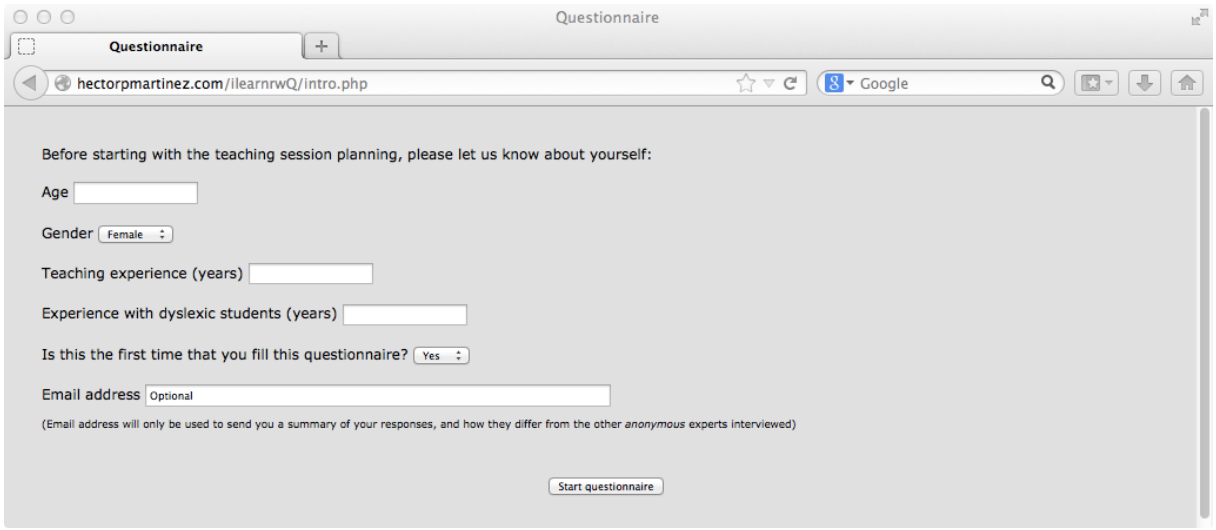


A.7. Answer format

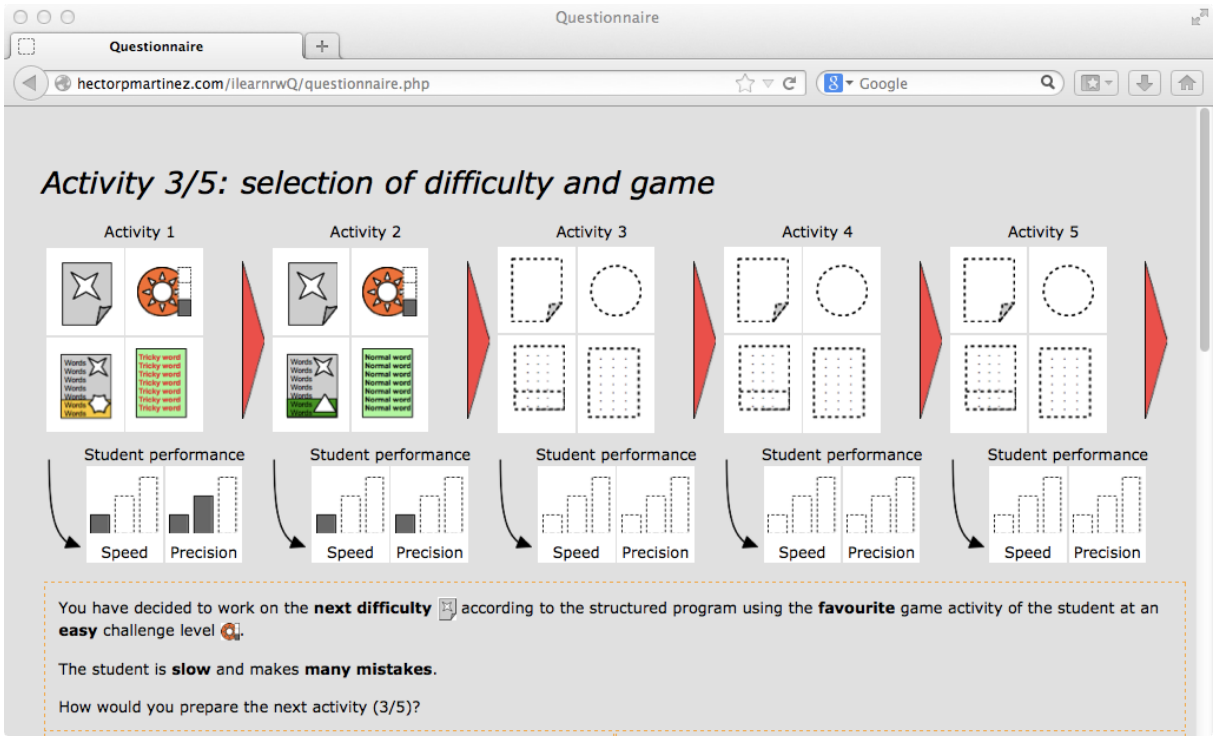


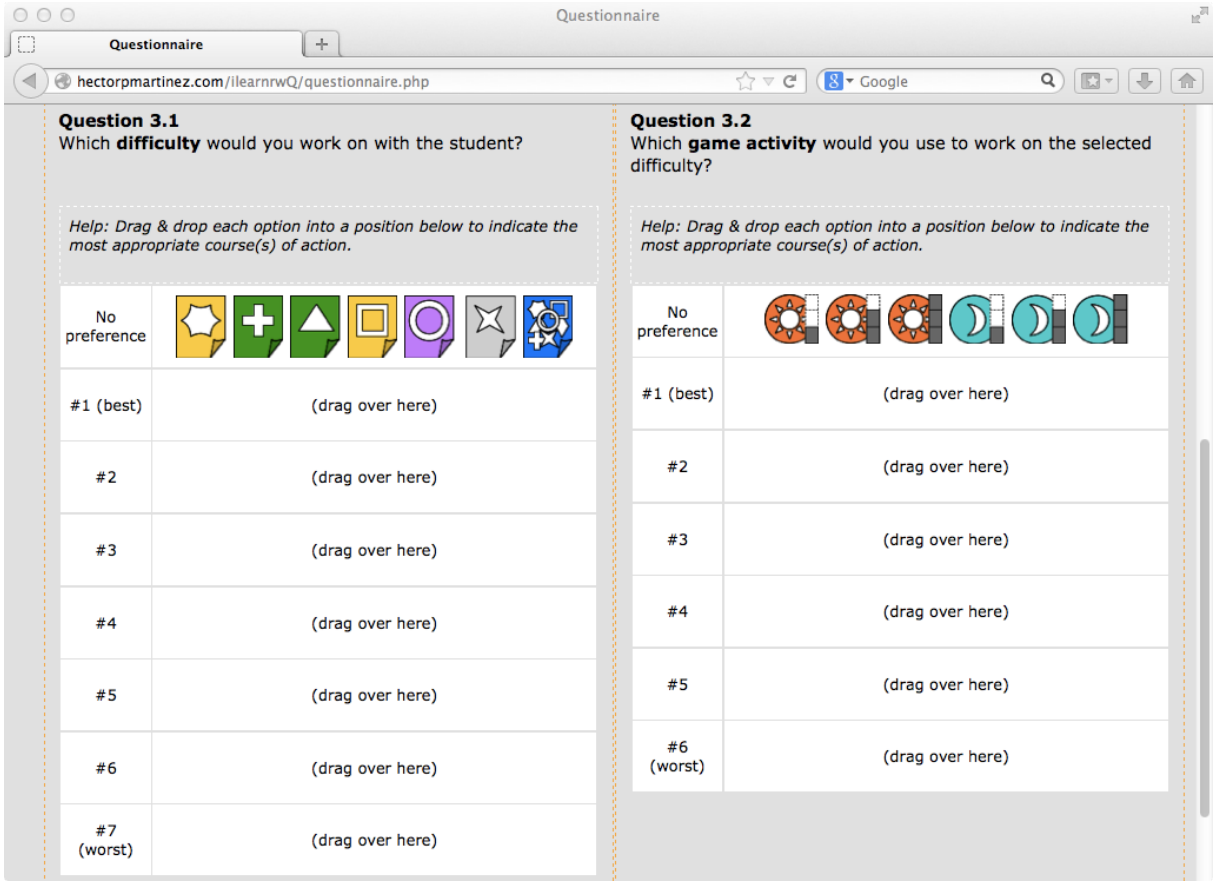


A.8. Demographics

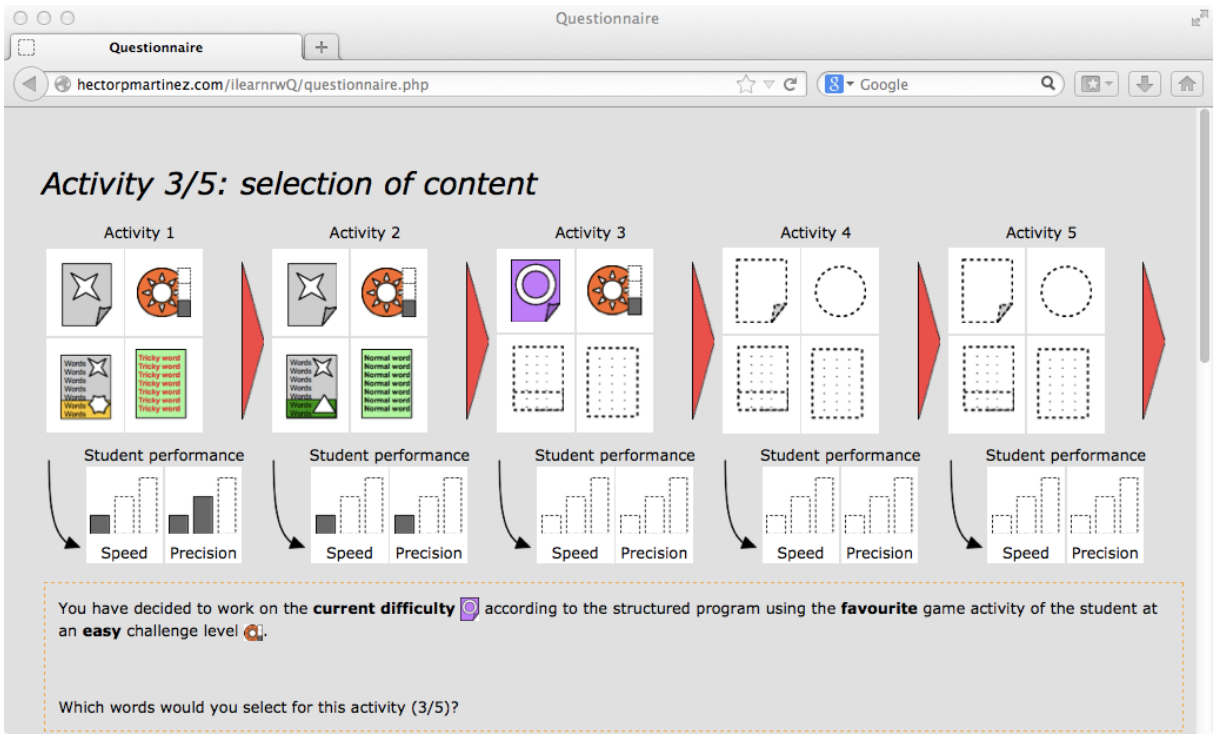


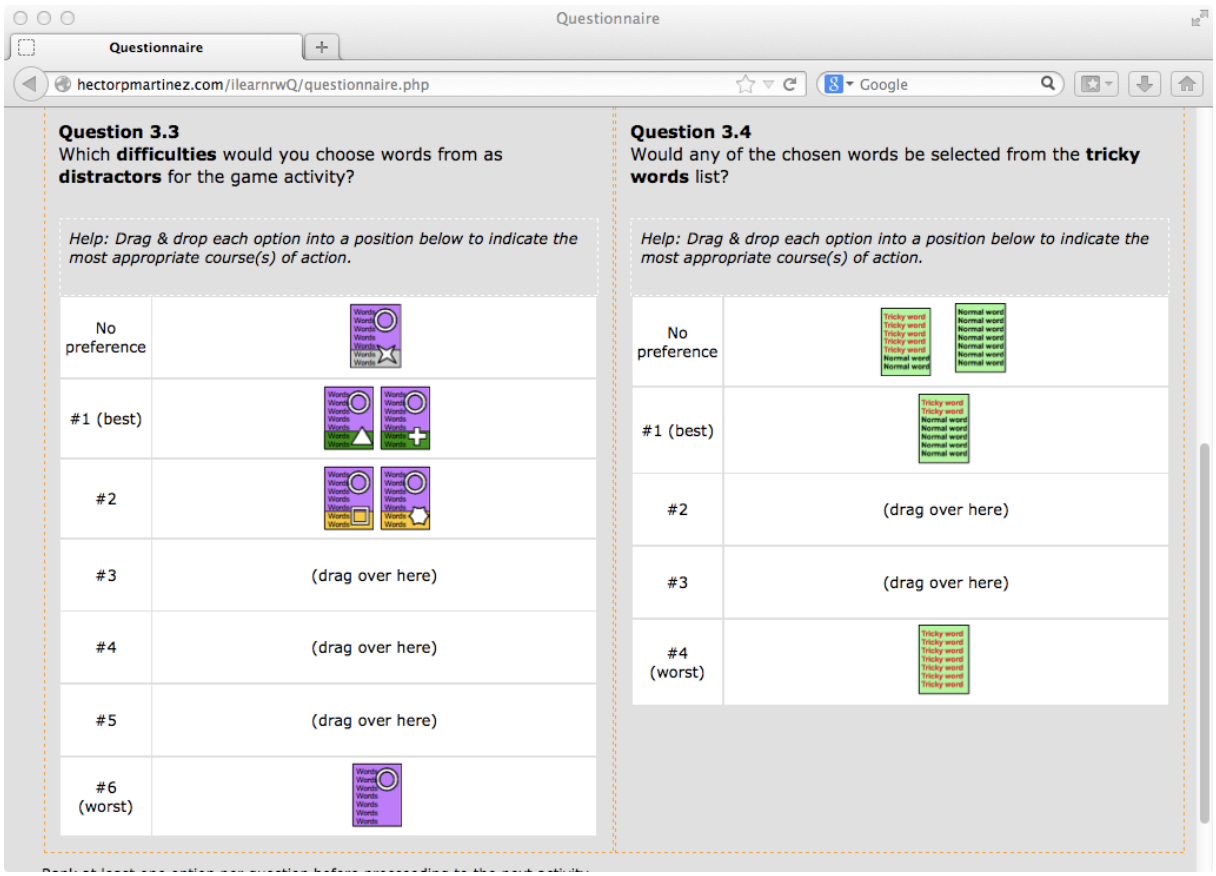
A.9. Question example: difficulty and game





A.10. Question example: content





Appendix B: DT preference learning pseudo-code

Algorithm 1: Algorithm to train a decision tree using pairwise preferences

```
01: expand node(F, n, r,P)
02:   if F is empty or P is empty then
03:       make node n a leaf node with rank equal to r
04:       return r + 1
05:   else
06:           τ* ← -1, f* ←{}, C* ← {}
07:       for every feature f in F and possible ordered value split C do
08:           calculate τ for the pairs in P if sorted by feature f using criterion C
09:           if τ > τ* then
10:               τ* ← τ, f* ← f, C* ← C
11:           end if
12:       end for
13:       make node n a non-terminal node with feature f*
14:       for c in C* do
15:           create new node n(f* →c) by creating branch from n using condition c
16:           P (f* →c) ← pairs in P where both samples satisfy the {f*,c} condition:
              {P(f* →c) is the list of pairs with unknown preferences}
17:           r ← expand node(F − {f}, n, (f* →c), r)
18:       end for
19:       return r
20:   end if
```

Bibliography

Abbasnejad, E., Bonilla, E., & Sanner, S. (2011). Sparse Gaussian Processes for Learning Preferences.

Aleven, V., McLaren, B., Roll, I., & Koedinger, K. (2004). Toward tutoring help seeking. In *Intelligent Tutoring Systems* (pp. 227-239). Springer Berlin Heidelberg.

Anderson, J., Boyle, C., & Yost, G. (1985). The Geometry Tutor. In Proceedings of the Ninth IJCAI, Los Angeles, Morgan-Kaufmann, San Mateo, Calif.

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *The journal of the learning sciences*, *4*(2), 167-207.

Bahamonde, A., Díez, J., Quevedo, J. R., Luaces, O., & del Coz, J. J. (2007). How to learn consumer preferences from the analysis of sensory data by means of support vector machines (SVM). *Trends in food science & technology*, *18*(1), 20-28. Chicago

Baker, R. S.d., Corbett, A. T., Koedinger, K. R., Evenson, S., Roll, I., Wagner, A. Z., ... & Beck, J. E. (2006). Adapting to when students game an intelligent tutoring system. In *Intelligent Tutoring Systems* (pp. 392-401). Springer Berlin Heidelberg.

Baker, R. S. (2007). Modeling and understanding students' off-task behavior in intelligent tutoring systems. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 1059-1068). ACM.

Beck, J., Stern, M., & Haugsjaa, E. (1996). Applications of AI in Education. *Crossroads*, *3*(1), 11-15.

Bloom, B. S. (1984). The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational researcher*, 4-16.

Brusilovsky, P., Schwarz, E., & Weber, G. (1996). ELM-ART: An intelligent tutoring system on World Wide Web. In *Intelligent tutoring systems* (pp. 261-269). Springer Berlin Heidelberg.

Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., & Hullender, G. (2005). Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning* (pp. 89-96). ACM.

Capuano, N., Marsella, M., & Salerno, S. (2000). ABITS: An agent based Intelligent Tutoring System for distance learning. In *Proceedings of the International Workshop on Adaptive and Intelligent Web-Based Education Systems, ITS*.

Chu, W., & Ghahramani, Z. (2005). Preference learning with Gaussian processes. In *Proceedings of the 22nd international conference on Machine learning* (pp. 137-144). ACM.

Conati, C., & Zhou, X. (2002). Modeling students' emotions from cognitive appraisal in educational games. In *Intelligent tutoring systems* (pp. 944-954). Springer Berlin Heidelberg.

D'Mello, S. K., Lehman, B., & Graesser, A. (2011). A motivationally supportive affect-sensitive autotutor. In *New perspectives on affect and learning technologies* (pp. 113-126). Springer New York.

Eliot, C. & Woolf, B. (1995). An Adaptive Student Centered Curriculum for an Intelligent Training System. User Modeling and User-Adapted Interaction, 5, 1, pp. 67-86.

Evens, M. W., Chang, R. C., Lee, Y. H., Shim, L. S., Woo, C. W., Zhang, Y., ... & Rovick, A. A. (1997, March). CIRCSIM-Tutor: An intelligent tutoring system using natural language dialogue. In *Proceedings of the fifth conference on Applied natural language processing: Descriptions of system demonstrations and videos* (pp. 13-14). Association for Computational Linguistics.

Fürnkranz, J., & Hüllermeier, E. (2010). *Preference learning* (pp. 789-795). Springer US.

Graesser, A. C., Lu, S., Jackson, G. T., Mitchell, H. H., Ventura, M., Olney, A., & Louwerse, M. M. (2004). AutoTutor: A tutor with dialogue in natural language. *Behavior Research Methods, Instruments, & Computers*, *36*(2), 180-192.

Grafsgaard, J. F., Wiggins, J. B., Boyer, K. E., Wiebe, E. N., & Lester, J. C. (2013). Automatically Recognizing Facial Indicators of Frustration: A Learning-Centric Analysis. In *Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on* (pp. 159-165). IEEE.

Haugsjaa, E. & Woolf, B. (1996). 3D Visualization Tools in a Design for Manufacturing Tutor. In Proceedings of Educational Multimedia and Hypermedia, Boston, Mass.

Joachims, T. (2002). Optimizing search engines using clickthrough data. In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 133-142). ACM.

Kendall, M. G. (1938). A new measure of rank correlation. *Biometrika*.

Knight, Y., Martínez, H. P., & Yannakakis, G. N. (2013). Space Maze: Experience-Driven Game Camera Control. in *Proceeding of Foundations of Digital Games*

Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. A. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education (IJAIED)*, *8*, 30-43.

Lajoie, S. & A. Lesgold (1992). Apprenticeship Training in the Workplace: Computer-Coached Practice Environment as a New Form of Apprenticeship. In Intelligent Instruction by Computer, Farr and Psotka, eds., Taylor & Francis, Washington, DC, pp. 15-36.

Martínez, H. P., Bengio, Y., & Yannakakis, G. N. (2013). Learning deep physiological models of affect. *Computational Intelligence Magazine, IEEE*, *8*(2), 20-33.

Martínez, H. P., Yannakakis, G. N. & Hallam, J. (2014). Don't Classify Ratings of Affect; Rank them!. *IEEE Transactions on Affective Computing.* (accepted with revisions)

Mitchell, T. M. (1997). Machine learning. 1997. *Burr Ridge, IL: McGraw Hill*, *45*. Chicago

Nielsen, J. B., Jensen, B. S., & Larsen, J. (2011). On sparse multi-task gaussian process priors for music preference learning. In *NIPS Workshop on Choice Models and Preference Learning*.

Ovadia, S. (2004). Ratings and rankings: Reconsidering the structure of values and their measurement. *International Journal of Social Research Methodology*, *7*(5), 403-414.
Pedersen, C., Togelius, J., & Yannakakis, G. N. (2009, September). Modeling player experience in super mario bros. In *Computational Intelligence and Games, 2009. CIG 2009. IEEE Symposium on* (pp. 132-139). IEEE. Chicago

Quinlan, R. (1986). Induction of decision trees. *Machine Learning*. 1(1):81-106.

Radlinski, F., & Joachims, T. (2005). Query chains: learning to rank from implicit feedback. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining* (pp. 239-248). ACM.

Robison, J., McQuiggan, S., & Lester, J. (2009). Evaluating the consequences of affective feedback in intelligent tutoring systems. In *Affective Computing and Intelligent Interaction and Workshops, 2009. ACII 2009. 3rd International Conference on* (pp. 1-6). IEEE.

Rokach, L. (2008). Data mining with decision trees: theory and applications (Vol. 69). World scientific.

Shute, V., Glaser, R. & Raghaven, K. (1989). Inference and Discovery in an Exploratory Laboratory. Learning and Individual Differences, Ackerman, P., R. Sterberg, and R. Glaser, eds., pp. 279-326.

Shute, V. J. (1993). A macroadaptive approach to tutoring. *Journal of Artificial Intelligence in Education*.

Shute, V. J. (1995). SMART: Student modeling approach for responsive tutoring. *User Modeling and User-Adapted Interaction*, *5*(1), 1-44.

Spronck, P., Sprinkhuizen-Kuyper, I., & Postma, E. (2004). Difficulty scaling of game AI. In Proceedings of the 5th International Conference on Intelligent Games and Simulation (GAME-ON 2004) (pp. 33-37).

Stern, M., Beck, J. & Woolf, B. (1996). Adaptation of Presentation and Feedback in an Intelligent Mathematics Tutor. In Proceedings of the Third International Conference on Intelligent Tutoring Systems, Montreal.

Vanlehn, K., Lynch, C., Schulze, K., Shapiro, J. A., Shelby, R., Taylor, L., ... & Wintersgill, M. (2005). The Andes physics tutoring system: Lessons learned. *International Journal of Artificial Intelligence in Education*, *15*(3), 147-204.

Weber, G., & Mollenberg, A. (1994). ELM-PE: A Knowledge-based Programming Environment for Learning LISP.

Winkels, R., Sandberg, & Breuker, J. (1990). The Coach. In EUROHELP: Developing Intelligent Help Systems, Breuker, J., ed., EC: Copenhagen, pp. 119-146.

Yannakakis, G. N., & Hallam, J. (2011). Rating vs. preference: a comparative study of self-reporting. In *Affective Computing and Intelligent Interaction* (pp. 437-446). Springer Berlin Heidelberg.