Universidad Carlos III de Madrid
www.uc3m.es

TESIS DOCTORAL

Contributions to affective learning through the use of data analysis, visualizations and recommender systems

Autor:

Derick Antonio Leony Arreaga

Director:

Dr. Pedro José Muñoz Merino

DEPARTAMENTO DE INGENIERÍA TELEMÁTICA

Leganés, febrero de 2016

TESIS DOCTORAL

Contributions to affective learning through use of data analysis and visualizations

Autor: *Derick Antonio Leony Arreaga*

Director: Dr. Pedro José Muñoz Merino

Firma del Tribunal Calificador:

Firma:

Presidente

Vocal

Secretario

Calificación:

Leganés, ___ de _______________ de _____

To my daughter, Mia
A mi hija, Mia

Agradecimientos

Siempre me he considerado una persona muy afortunada, en especial por las personas que he conocido y que me han ofrecido su ayuda, apoyo y ánimo. Esta tesis doctoral es otro ejemplo de los frutos de la ayuda constante de dichas personas, a las que les estoy muy agradecido.

En primer lugar, le agradezco a mi director, Pedro J. Muñoz Merino, por su constante guía, consejos, aportaciones y ánimos. Todo esto, en especial durante la recta final, ha sido esencial para concluir este proyecto. También le a agradezco a Abelardo Pardo, quien dirigió inicialmente esta tesis y cuyos consejos y aportes sirvieron para sentar mis líneas de investigación. Tanto a Pedro y Abelardo como a Carlos Delgado Kloos, les agradezco haberme hecho sentir como en casa desde mi incorporación a la universidad en el 2007, y por sus consejos y gestiones para permitir mi desarrollo académico y profesional.

Gran parte del conocimiento y metodologías aplicadas en este doctorado son el resultado de haber trabajado con gente maravillosa, integrantes del grupo Gradient del Departamento de Ingeniería Telemática de la Universidad Carlos III de Madrid. Gracias a Luis, Mari Carmen, Mario, Raquel, Julio, Isra, María Blanca, Mar, Carlos Alario, Hugo, José Jesús, Iria y José.

Le agradezco a mis compañeros de despacho, primero en el 4.1.C03 y luego, con mayor integración por compartir las alegrías y penas de una tesis doctoral, en el 4.1.F05: Isra, gracias por compartir proyectos tan interesantes, *summer schools* y demás anécdotas; Alicia y Damaris, gracias por tantos buenos momentos.

Mi agradecimiento se extiende también al resto de integrantes del grupo GAST y del Departamento de Ingeniería Telemática. En especial a aquellos con los que compartíamos el café y la comida, en los que mezclábamos momentos para desconectar como para resolver temas de trabajo. Gracias Estrella, Patri, Gustavo, Edu, Dani, Alberto, Gordillo, Iago, Pablo, Diego, David, María, Carlos G., Celeste, Andrés, Florina, Jesús, Norberto y Grego.

I also want to thank my hosts during two research stays conducted as part of my doctorate. My research at Fraunhofer Institute for Applied Information Technology in Bonn with the CAPLE group of Martin Wolpers was a key experience to set the initial model of my thesis. My second stay, at WISE Research Group of Vrije Universiteit Brussel and supervised by Katrien Verbert, allowed an important advance of my thesis research. Thank you all for your hospitality and help.

Finalmente, doy gracias a mi familia y amigos, quienes siempre me ha apoyado en mis inquietudes académicas. Gracias a mis padres, Homero y Dina, y a mis hermanos Raúl y Lizz por su constante apoyo. Muchas gracias a María Alejandra, quien ha vivido a mi lado los momentos dulces y los no tan dulces de un doctorado, y que con su amor me sigue demostrando lo afortunado que sigo siendo.

Abstract

Student modeling is an important issue in telematics learning environments, e.g. learning resources can be adapted based on the students' information. An emergent area of student modeling is the inclusion of affective information. The improvement of emotion detectors based on the students' events in different telematics learning environments is an open issue. Moreover, there is a need of proposing and evaluating new visualizations involving affective information, and proposing generic solutions for the recommendation of learning materials based on the affective information.

This PhD proposes two different models for the detection of emotions in two different telematics learning environments. The first model uses a Hidden Markov Model to infer the emotions in a programming learning environment in which students should use different tools to learn how to program. The second model uses a set of rules to infer the emotions in a Massive Open Online Course platform in which students should solve exercises and watch videos.

An evaluation of the first model for the detection of emotions was performed using a controlled experiment, comparing the results of the model with the students' answers regarding their emotions in different instants of times. The results showed that the model was not able to detect accurately the students' answers regarding their emotions. Other models of the literature applied in other learning environments were tested and they were not able to predict accurately the students' answers regarding their emotions. Therefore, the detection of emotions based on students' events in these types of environments might not be feasible, or the reference data of students' answers to a survey with different questions about emotions should be redefined.

Moreover, this PhD proposes a set of affective-related visualizations for learning environments. Some of these visualizations only involve affective information, while others combine this affective information with other related to the students' activities with the learning platforms. Some of these visualizations were evaluated with real students and

results showed a good usability, usefulness and effectiveness.

Finally, this work proposes a generic framework for enabling the recommendation of learning resources based on affective information. The solution includes an Application Programming Interface for the definition of the different possible events. A specific implementation of this recommender has been developed as a plugin of the ROLE SDK platform.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Learning analytics can be defined in a broad way as the science for the collection, transformation, analysis and interpretation of educational data with the purpose of improving the learning process [1].

In the last years, the popularity of learning analytics has increased with the emergence of big amounts of educational data in platforms such as MOOCs and with the possibilities of new applications. From 2011 the series of conferences about learning analytics started with LAK 2011[1]. Moreoever, there was an intensive previous of research in related areas such as educational data mining [2].

The different phases of the learning analytics process can be explained as a life cycle, in which first the data is collected from different educational environments, next this raw data can be transformed and combined with other information into useful information and finally this information can be used by different actors (teachers, students, managers, etc.) through different actuators, such as visualizers, recommenders or generators of reports [1].

The data that are generated by the learning environments include students' events in the platform about the user, the type of actions, the time, or the application where the action took place. This data are in the form of logs and each event is stored as a log. This data are meaningless by themselves. But from these data many things might be inferred. A selection of important aspects that can be inferred is the following:

1. Students' cognitive states. The knowledge level of each student in different skills can be inferred and their effectiveness and efficiency with the different educational

[1]https://tekri.athabascau.ca/analytics/

materials.

2. Students' meta-cognitive states. The meta-cognitive skills of the students can be inferred such as the help-seeking behavior or their capacity for setting goals.

3. Students' behaviors and preferences. Behavioral patterns and preferences might be inferred from their actions.

4. Students' emotions. Different types of students' emotions might be inferred.

5. Students' engagement and motivation.

There are very diverse learning environments with their own characteristics such as Learning Management Systems (LMSs), MOOCs ad-hoc environments for using specific tools to be learnt, or Intelligent Tutoring Systems (ITSs). Depending on the learning environment that is used, the inference of some indicators might be possible or not, and the way of inference depends on the semantics of such environment.

1.1 Motivation

The emotions and affective states that a person experiences have implications in the way they think, behave and act. The reverse relationship is also true in the sense that the way a person thinks and behaves as well as their own experiences lived have an implication in their emotions and affective states. The educational process implies students' actions and behaviors so the students' affective states will have implications in the learning process.

For example, if a student is frustrated, he/she can answer incorrectly a set of questions that he/she would know in normal conditions, and he/she can do inappropriate actions in the system. Or if a student is very happy, he/she can do his/her best and answer correctly a difficult question for him and do the proper steps in the platform to get the correct solution. Therefore, the students' affective states have implications in the way how students learn and behave in educational systems.

Theories of self-awareness emphasize the importance of the students to know about their own features during the learning process and interpret them to act accordingly [3]. In this context, it is important that students can be aware of their own emotions, their evolution and their relationship with other important indicators of the learning process.

In addition, it is also important that teachers and other stakeholders can track and be aware of their students individually, but also as groups and as a whole, so that they can take the proper decisions during the learning process. The inclusion of affective information in these decisions is a key issue as the way how students learn can be considerable improved if the affective information is known and teachers make proper decisions taking also into account pedagogical implications.

Adaptive learning is also a well known strategy in which contents are adapted and personalized to each student depending on their features and characteristics [4]. The adaptation is done based on user models that can consider a lot of variables related to the students. One important aspect that can be considered for the adaptation of the contents is the emotions and affective states. As the students acts and behaves differently depending on the emotions they experience, the way they learn can be adapted considering these affective states to improve the learning process.

Therefore, an important issue is how to detect the students' emotions and affective states in each moment. First attempts to do this were based on students' surveys where students had to fill in their emotions. This method might not be reliable enough because some students can cheat or not knowing their affective states. In addition, other approaches are based on different sensors that capture parameters such as the body temperature, the pressure or the EGM signals. These approaches are too intrusive as they require that students must have different devices in their bodies when they are learning. Finally, there are approaches that try to detect the students' emotions just based on the own actions of the student with the system. Different detection methods have already been proposed for Intelligent Tutoring Systems (ITSs) but much more effort is required to propose detectors for other learning environments such as Massive Open Online Courses (MOOCs) or ad-hoc environments for learning different tools by practice (e.g. programming environments with the use of the compiler, editor, memory checking tools, etc.). One of the challenges is the accuracy of these detectors in order to try to detect the students' emotions, since depending on the environments this information might be difficult to infer because of the semantics of the learning environment.

Moreover, useful, effective and usable visualizations should be provided to students and teachers about the emotions and their relationship with other indicators. There is a need of proposal of a set of visualizations related to emotions and their evaluation to validate them.

In the direction of adaptive learning, the design and implementation of recommenders enable to give personalized educational resources depending on the students' profiles. Although there are generic architectures that can be applied for personalized learning and recommendations, much more can be done for the definition of specific architectures that take into account the affective information but being generic enough to be applied to different scenarios with the use of affective information.

1.2 Objectives

In this dissertation, we aim at making several contributions in different phases of the whole life cycle of learning analytics applied to emotions and affective states. Specifically, the objectives of this dissertation are the following:

1. Provide detection models of emotions and affective states in different learning environments using only the information of the students' events with the learning environments (e.g. without use of external devices for capturing the face or body signals) and implement detectors of these detectors in the considered learning environments:

   (a) Provide detection models and implement detectors of emotions and affective states in a learning environment in which students should use different tools that they should learn and interpret. These tools (their use, analysis and interpretation) are part of the learning process.

   (b) Provide detection models and implement detectors of emotions and affective states in a MOOC learning environment with a high student interaction with videos and exercises. The contents of the learning activities should be learnt but not the use of the activities themselves, as difference with the previous learning environment.

2. Evaluate the proposed detection models of emotions and affective states of the previous point to know if they are accurate for the detection of the students' emotions and affective states, and compare the proposed detection models with other detection models of the literature based only on students' events that worked well in other different learning environments.

3. Propose a set of visualizations related to emotions and affective states, which are not connected to a specific learning environment, which might be useful for the learning process.

4. Evaluate the usefulness, usability and effectiveness of the proposed visualizations about emotions and affective states presented in the previous point.

5. Propose and implement a framework for the recommendation of learning resources based on emotions and affective states.

Related to the presented learning objectives, there are the following research questions:

- Can detection models of emotions and affective states be provided based on events of students for different learning environments based on pedagogical background, good practices, previous expertise, etc.?
- Can the proposed models of emotions and affective states proposed be accurate enough to really detect the students' states?
- Can general visualizations involving emotions and affective states be proposed to give nice insights about the learning process?
- Are the proposed visualizations related to emotions and affective states useful, usable and effective?
- Can we provide a general API and recommendation framework to manage emotional and affective students' states?

1.3 Structure of this document

The document is divided into seven chapters, which are described as follows:

- CHAPTER 1: INTRODUCTION. This is the present chapter in which the motivation is presented as well as the objectives of this dissertation.
- CHAPTER 2: RELATED WORK. This chapter makes a review of the different methods of detection of emotions in learning environments, the accurate of this

detection, visual analytics tools for learning environments, and recommenders in technology enhanced learning.

- CHAPTER 3: DETECTION OF EMOTIONS IN NEW LEARNING ENVIRONMENTS. This chapter proposes two different models of detection of emotions based only on the students' events for two different learning environments for which the detection of emotions was not previously covered: a programming learning environment and a MOOC based one.

- CHAPTER 4: VISUALIZATIONS OF EMOTIONS IN LEARNING ENVIRONMENTS. This chapter makes a contribution with the proposal of different visualizations related to emotions and affective states and their relationship with other important learning indicators.

- CHAPTER 5: EVALUATION. The evaluation will be divided into three main topics: evaluation of the accuracy of the detection of emotions in one learning environment, evaluation of the relationship of the affective information with some relevant learning indicators and the evaluation of some of the proposed visualizations.

- CHAPTER 6: RECOMMENDATION BASED ON AFFECTIVE INFORMATION. This chapter proposes a recommendation framework for the use of affective information, including the definition of an API.

- CHAPTER 7: CONCLUSIONS AND FUTURE WORK. The last chapter will include some conclusions, the related publications and research projects to this dissertation and possible future works.

Chapter 2

Related work

2.1 Student modeling

The topic of student modeling has been worth of extensive research in the last decades. A student can be modeled based on very different features and characteristics. Initial research on student modeling focused mainly on skill modeling of students while other features such as emotions, engagement or gaming the system have been included more recently. A book that reviews student modeling from different points of view can be found in [5].

Modeling of students can be used for different purposes. For example, teachers can provide better feedback to their students, students can be aware of themselves or automatic personalized systems can be implemented that adapt the learning resources depending on the specific students' features and needs. An overview of Adaptive Hypermedia Systems with different possible techniques of adaptation is presented in [6]. Different personalization and adaptation frameworks and systems have been presented in the last years [7, 8, 9, 10, 11].

A analysis of different aspects that can be modeled in data mining and education is presented in [12]. A classification of the different aspects to be modeled is provided next:

- Cognitive skills. A skill can be defined such as anything that a student needs to have a certain performance. A skill can be a specific topic, misconception or procedure among others. The probability to know a specific skill for each student can be identified.

- Meta-cognitive skills. These skills are related to know certain strategies about how to learn. Engagement and motivation. This is the commitments to have a specific behavior based on several factors. Affective states and emotions. These are the feelings that a person experiments such as happiness, frustration, angry, surprised or scared.

- Engagement and motivation. These are stats in which learners are interested on the subject they are studying and eager to work on the topic.

- Learning styles. The learning styles are based on the concept that different students differ in the way they learn and that people can learn better using a specific set of features or procedures. For example, one type of learning style might be related to the type of resources students can use to learn better, classifying students among visual, auditive or kinesthetic.

- Gaming the system. This is a student attitude to try to obtain a good final result in a learning environment but instead of a process of learning as a result of taking advantage of some system vulnerabilities.

- Social interactions. The way a student interacts with other peers and teachers in order to learn. These social interactions might usually include tools such as forums or chats.

- Other behaviors. Any other behaviors in learning platforms can be modeled such as the way of interaction with gamification elements.

Each of these aspects is analyzed in each of the following subsections while the affective states are covered in detailed in the next section.

2.1.1 Cognitive skills

Different approaches have been taken for the modeling of cognitive skills. A review of these approaches can be found at [13]. Some of the more usual are the following:

- Item Response Theory (IRT). This probabilistic model defines a unique latent trait, which is a variable representing the student knowledge level of a topic as a whole [14]. This is a continuous variable. Therefore, there is only one skill in this model instead of a division in skills. In addition, there is an Item Characteristic

Curve (ICC) for each question that provides the probability that a student solve the question correctly depending on the student knowledge level. This ICC curve can be only defined by one parameter (the difficulty of the item), two parameters (the difficulty and the slope) or three parameters (the difficulty, the slope and the probability of guess). Depending on the students' answers to the questions, the skill latent trait is updated. The IRT method allows the estimation of the student knowledge in a skill as a whole. The IRT methodology has been applied in works such as [15] or [16].

- Bayesian Networks (BN). The bayesian networks are graph models in which the nodes represent random variables. In the case of skill modelling these nodes usually represent the student knowledge level in different skills or the answer of students to different questions. The nodes usually represent binary variables, being e.g. 0 as not mastering a skill and 1 as mastering a skill; or 0 as answering a question incorrectly and 1 as answering a question correctly. In addition, the bayesian networks include arcs which represent a relationship between two nodes of the graph. The graph of a bayesian network must be acyclic. A property of a bayesian network is that the probability density function of a variable is independent of the other variables of the network if the value of the parent node variables are known. In order to define a bayesian network, the prior probability of the nodes without parents should be provided and the conditional probability tables of each node with parents should be provided. A good review of bayesian networks applied to learning environments can be found at [17]. The bayesian networks have been applied in different e-learning systems for the detection of skills based on the evidence of answering different questions. Some examples include Andes [18] in a fine-grained modeling or in Assisstment [19] in a less fine- grained modeling.

- Knowledge Spaces (KS). The most known system that uses the Knowledge Spaces theory is ALEKs [20]. The Knowledge Spaces theory does not define the specific skills but just the different items are represented in a graph and there are blocking relationship, so if an item is not mastered by a student, the student is not able to go to the following items of the graph. Some approaches including skills in the Knowledge Spaces theory have been proposed such as [21] in which definition of competencies are established.

- Deterministic models. Some models are not based on probabilistic models but on deterministic models such as [22] that takes into account the relationship among items in order to calculate the effectiveness. Approaches based on semantic web techniques usually take deterministic techniques for the calculation of the different skills and these skills are incorporated in an ontology and can be used as part of the decisions for adaptation [23].

2.1.2 Meta-Cognitive skills

Meta-cognitive skills are skills about how to acquire cognitive skills. Therefore, it can be viewed as the skills of a student for learning a course, concepts, etc. in a proper way. Some examples of meta-cognitive skills are the capacity of a student of self-awareness and recognize errors in his/her learning process, help-seeking behavior as the capacity of a student for requesting for help when he/she needs it, or the capacity of a student for setting goals.

In some learning environments some of the meta-cognitive skills can be detected such as the help seeking behavior or unreflective users [24, 25].

2.1.3 Engagement and motivation

Motivation and engagement are key in learning environments because if students are motivated and engaged, they might learn better and they will have a desire to interact and complete activities in the platform.

A good review of measuring of engagement in learning environments is presented in [26] where the calculation of engagement in the literature is divided on different indicators in different categories: behavioral, cognitive and emotional.

Different proposals for measuring motivation have been proposed such as in [27].

2.1.4 Learning styles

Learning styles affect the way of how students learn [28]. This has implications in e-learning systems since e.g. we can try to adapt the learning materials according to the student features and characteristics. For example, a student with a visual learning style might receive videos instead of audios. Or for example, a student with collaborative learning style might receive more collaborative activities during the learning process.

There are a big amount of learning styles that can be defined according to different dimensions. The work [29] presents a global review of the last years about learning styles.

Specific implementations of systems that recognize and use learning styles have been proposed such as [30] or [31].

2.1.5 Gaming the system

Some students try to overcome the rules of learning systems in order to game the system, so that e.g. these students can get a good scoring result in the system even if they do not know the topics covered but just because they know how to exploit some system capabilities, e.g. answering a multiple choice question several times until the system gives them the correct answer.

Different works have implemented detectors of gaming the system for students such as [32].

2.1.6 Social interactions

The social interactions of students with learning platforms are very important since they can give very useful information such as which students are connected, if students liked to answer other peers, etc.

A proposal of open social model for learning environments has been proposed [33].

2.2 Emotion detection in learning environments

According to different theories, the actions of people are a result of their affective states. Using technological means in order to infer the current affective state of a person is part of the field known as affective computing [34].

The detection of emotions in learning environments can be done in very different ways looking at their attitude and behavior. The following elements have been used for the detection of emotions in these scenarios:

- Cameras. The image of students can be used for detecting affective states, focusing on aspects such as face gestures or their posture. Initial studies in this area by Ekman [35] concluded the features related to each of the denominated basic emotions (happiness, disgust, fear, surprise, anger and sadness). An example of these types of methodologies for detecting emotions is presented in [36].

- Audio. Microphones have added a input channel to infer emotions in technological environment. Sounds like laughter, screams and their voice characteristics, like pitch, tone and volume provide relevant background information of the locutor state. An example of the analysis of audio to detect affective state is provided in [37].

- Physical sensors. Sensors to measure heart rate or skin conductance have been used to help in the detection of emotions in learning environments [38]. Another kind of sensors focuses on the pressure applied to any device while learning; common examples are senors on the computer mouse to detect the strength of the grasp and sensors on the chair to detect the learner's posture.

- EEG signals. EEG signals are voltage signals which can be measured on the scalp and give insights about the neuronal activity of the learners. Some examples of use of these techniques can be found in [39] and [40].

The use of physical sensors, cameras, audio or EEG signals require an intrusive methodology in which students should be put with different devices during the learning experience. In addition, it might not be feasible to have the different devices available to be used in all the learning experiences. For these reasons, other non-intrusive methodologies might improve these practical aspects of the detection of emotions even at the price of losing accuracy in the prediction. The detection of emotions based on the learner's actions is a prominent area.

Research work related to the detection of emotions from learner's actions in educational environments can be analyzed from three perspectives: 1) inferring emotions from observations in computer-supported educational settings; 2) studies on the relationship between learner's emotions and their performance; and 3) research related to the analysis of behavior in an educational programming environment. We provide a summary of prior studies that fit into each perspective.

The first perspective includes studies with the goal of detecting emotions of a learner from the analysis of her actions while using an educative computer application. Most of the studies in this area involve the use of educational games or ITS. Sabourin et al. [41] define a dynamic Bayesian network for the detection of emotions in the game-based learning environment Crystal Island. When compared to a baseline, the accuracy of their model was 10 points above in emotion detection and 18 points in valence accuracy. For the case of ITS, D'Mello et al. [42] have worked on the prediction of emotions from dialogs in the ITS AutoTutor. They were able to recognize confusion, eureka and frustration emotions based on the analysis of the dialogue maintained with the student.

Another study on the line of ITS was done by Arroyo et al. [43] where they used both a sensor and a sensor-less approach to detect student emotions while working with Wayang Outpost, a geometry ITS. Finally, Baker et al. [44] have worked on the detection of emotion in an algebra ITS called Cognitive Tutor Algebra 1. They were able to improve previous results in the recognition of engaged concentration, confusion, frustration and boredom.

The second perspective involves studies that analyze the relation between student's emotions and their performance in the learning activity. In this line of research, Craig et al. analyzed the correlation between a set of six emotions with the presence of learning. Their study concluded that confusion and flow correlate positively with learning gains, while boredom presents a negative correlation [45]. In a similar line, Baker et al. found that the emotion of boredom tends to make students to game the system, i.e. look for shortcuts to advance in the ITS [32]. The observation of this behavior had previously been associated with poorer learning [46]. More recently, Pardos et al. presented an approach to study the effect of emotions on scores of tests conducted at the end of the academic year. The results of this study agree with prior reports: engaged concentration and frustration correlate positively with the scores, while boredom and confusion correlate positively in scenarios with scaffolding and negatively without it [47].

Studies involved in the third perspective analyze the actions of students during a learning activity in the domain of computer programming. Lee et al. analyzed the logs generated by students learning to program the language Java to label whether the student was confused. They found a negative association of student achievement with prolonged confusion and a positive association with confusion that is resolved [48]. Another study of interest was conducted by Blikstein and consisted in the collection and analysis of a set of metrics of student's actions in NetLogo, a programming environment. The metrics analyzed included the size of the code, time between compilations, error rate, and compilation attempts [49].

2.3 Visualizations in technology enhanced learning

The use of graphs or visualizations to present information has been a topic of study during the last decades. One of the shortcomings of several educational systems is the difficulty for presenting powerful but at the same time easy to understand visualizations for the stakeholders about the inferences that are the result of certain processing of data. Research about how to obtain easy to use and interpret visualizations for involved stakeholders in these systems has been addressed in several works.

Visualizations related to the learning process are an important issue of study in recent years. Different works presented useful visualizations for various aspects in the field of technology enhanced learning. Examples of these visualizations include resource accesses over time [50], detailed information about the interactions with exercises and hints, activities on an LMS [51], number and types of events and items [52, 53, 54] or social interactions [55].

Mazza and Dimitrova were among the first to propose the application of information visualization to education with CourseVis [56] and then continued with the GISMO Tool [57]. The tool provided visualizations of behavioral data obtained from the logs of a Learning Management System (LMS). An instructor was able to see graphs reporting the students' accesses to the course (see Fig. 2.1), either as a time line or in a histogram. Another graph presented the discussion performed by a learner in the course and a visualization of the submission time for course assignments. Authors provided a case study to illustrate the application of GISMO Tool in the day-to-day activity of an online course. This tool has also been used by Petropoulou et al. as part of the Learning Analytics Enriched Rubric (LAe-R) [58] to assess learners' performance. A case study proved the tool to be usable and accepted by practitioners as it provided guidance by including performance indicators obtained from the analysis of learners' interactions and behavior.

Visualizations have also been often used in collaborative learning environments to present the interactions between all stakeholders and also between learners and educational resources. Juan et al. have proposed an information system model that generates weekly reports to monitor learners' and groups' activities [59]. Their system, called SAMOS, presents four visualizations: a group risk classification tool, an individual risk classification tool, an evolution of group activity, and an activity graph for group members. Authors have designed the visualizations so that individuals and groups with low

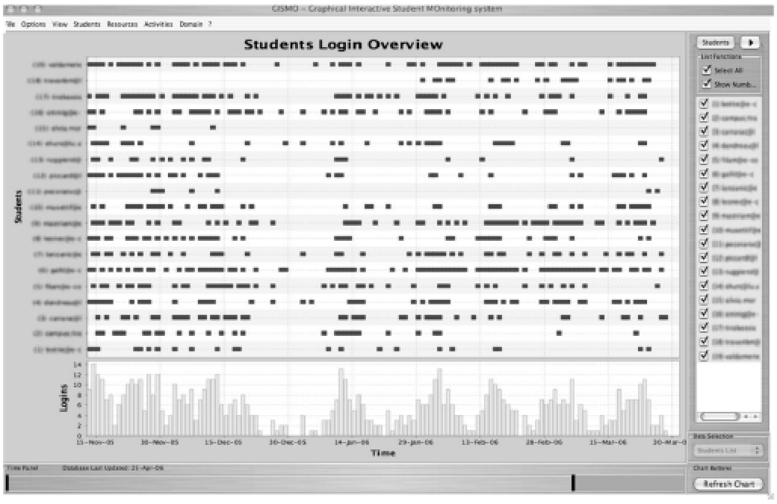

Figure 2.1: Students' access to the course report in GISMO.

activity levels are identified promptly. They also propose to use the tool as an indicator of potential problems which would allow an instructor to forecast collaboration issues and act accordingly to avoid them.

Also in the scope of collaborative learning, Brooks et al. proposed applying information visualization to identify aggregations of learners as part of a data-assisted approach to build intelligent TEL environments [60]. The authors argument how instructors can derive meaning from visualizations in order to improve their pedagogical practice. Their study represented the interaction of learners in a discussion forum through sociograms, graph-based structures where nodes symbolize a participant and arcs the a message between participants. In this format, nodes can be formatted to show distinct attributes, like amount of participation through the size or participant role through its color. Authors described use cases that applied this type of visualizations in three contexts: online small courses, face-to-face large courses, and comparison based on granularity.

Continuing with proposals of visualizations as early-warning systems, Mcfadyen and Dawson provided a proof of concept in which they obtained 15 variables that correlated highly with student achievement [61]. Among the results of their work they present the variables that had the highest correlation: total number of messages posted on the forum, number of finished assessments, and amount of mail messages sent. In addition, they present a set of network visualizations with a color-coded design to show the final grade of the learner with the color of the node that represents her.

On the same track, Dyckhoff et al. presented in [62] the design and implementation of the system eLAT (exploratory Learning Analytics Toolkit). eLAT allows teachers to explore user properties and behavior, including usage of learning objects. Among the requirements of the toolkit there are generic characteristics like usability, usefulness, extensibility and, reusability, plus properties tightly connected to large amounts of data: interoperability, real- time operations, and data-privacy. The visualizations presented in the toolkit are organized in a dashboard with a monitoring view with four widgets: usage, performance, behavior, and communication. Each of the widgets has a number of tabs that presents a specific visualization. The instructor is also able to configure the visualization through a configuration of filters in the analysis view.

Gaudioso et al. presented the positive results of providing instructors with a tool to monitor the activity and behavior of learners [63], mainly decreasing the drop rate and increasing the passing rate. Instructors were provided with a reporting and data analysis tool which included reports on predefined indicators, and reports with results of automated analysis. Predefined indicators included results of an initial questionnaire, individual profile, and group performance. The automated analysis consisted of clustering techniques to group learners according to data such as background, interaction patterns, and usage of learning resources. A capture of the reporting tool is shown in Fig. 2.2.

In addition to provide feedback to the instructor, visualizations have also been studied as a resource for self-assessment. An example is provided by Melero et al. by generating visualizations from logs of a mobile learning design [64]. Displayed information included the time dedicated by the learners to perform each activity, geographic information of routes followed by learners, the frequency in which each activity was done, and the score obtained during the activity. Their study concluded that visualizations were indeed able to highlight aspects of the learning design that could be improved. Moreover, learners that used the visualizations made a better diagnosis of their performance. Another conclusion of their work is the need to limit the amount of information provided to stakeholders for the visualizations not to generate a cognitive overhead during the activity.

Continuing on approaches that apply visualizations for self-assessment or self- reflection, Schmitz et al. introduced the tool CAMera to monitor and report on learning

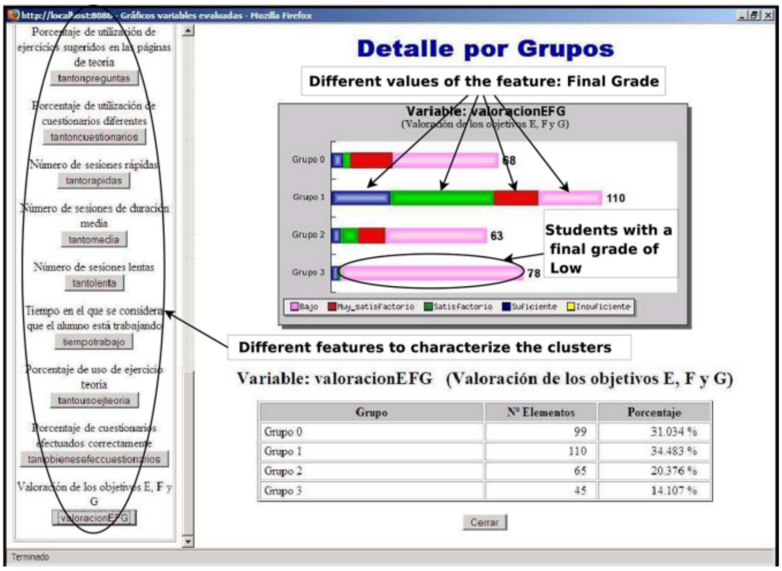

Figure 2.2: Capture of Reporting and Data Analysis Tool by Gaudioso et al.

behavior [55]. The tool collects behavior metadata from several applications, stores it and makes it available to the learner through two applications. First, an email-analyzer processes data related to receiving and sending emails; the interaction is then represented as a social network in which senders are shown as nodes and messages as arcs. The second application is MACE Zeitgeist, a set of web services that provide an overview of activities within a federation of architectural learning repositories.

The use of several graphs or visualizations as an overview of learning performance and activities status has been highlighted in studies focusing on learning dashboards [65]. Verbert et al. presented an exhaustive overview of these tools to support awareness, reflection, sense-making and learning impact [66]. The study concludes there is a need to assess the usefulness of dashboards and the information they present. There is also a need for completeness of data through a diverse set of sensors. Moreover, they highlight the complexity to evaluate these tools, their dependency on the learning setting and the surge of research communities following data-driven methodologies in educational technology.

The challenge of identifying information valuable by the stakeholders in learning activities has been approached by Stephens-Martinez et al. in [67]. They work focuses on identifying information that would allow instructors of MOOCs to understand the dynamics of their class. Thee authors obtained comments and insights from 92 instructors that were surveyed about their background information, details about a MOOC they had taught, course monitoring goals, and opinion on visualization mockups. Instructors' responses show a high use of discussion forums, class surveys, and direct feedback from students and assistants as the main resources for course monitoring. On the other extreme, quantitative information like pattern of students activity and chat room logs were used the least. Regarding the visualization mockups, instructors expressed their interest to use them while the course is running rather than to prepare new course material.

In addition to serve as an observation tool, visualizations can be used for other purposes. Using visualizations as a means of evaluation of educational resources is proposed by Krystalli et al. in [68]. Authors present the application of infographics, a specific subcategory of visualizations with appealing and communicative design, to present the evaluation of serious games for foreign language learning.

Visualizations have also been proposed as a pedagogical resource. For example, Chiu and Linn [69] propose the use of topic-specific visualizations to enhance an inquiry-oriented online curriculum. Their proposal is applied in the field of chemistry to connect concepts, symbols and molecular representations of chemical reactions. They used interactive dynamic visualizations of atomic molecular motion, provided by chemistry scene investigators. An empirical evaluation consisted on a pilot study that compared the performance of learners using the visualization- enhanced inquiry unit against learners following a traditional methodology. Their work presented positive results as learners that used visualizations showed a higher performance and were able to form more connections. On a similar approach, the application of visualizations as domain-specific resources, in this case to learn programming, is presented in [70]. The authors propose a conceptual framework to apply a visual representation of a program to better understand its control and data flow.

Other uses of visualizations can be mentioned, not fully centered in learning environment but still applicable in the field. For instance, an improved visualization method for self-organizing map is proposed for making easier to understand the related information [71], or some techniques have been applied for visualization of blogs [72].

Nevertheless, only few works have addressed the visualization of emotions in a learning context. Most of the existing works represent emotions as a color palette that matches each emotional state with a different color [73]. In addition, most works focus on the detection of emotions in texts based on semantic analysis [74]. In this line, Skynesketch [74] is a tool integrated in Moodle that recognizes emotions from text and is able to represent some emotions during time.

2.4 Recommendations in technology enhanced learning

Recommender systems are among the instruments used to provide learning environments with personalization. This occurs by suggesting learning resources that are most suitable for the characteristics and behaviors of the learner. For example, a learner that has accessed material to learn the basics of Spanish may be interested in other resources related to the subject. Later, she may benefit from being offered with material of intermediate Spanish. Manouselis et al. [75] describe the approaches followed so far to implement and deploy recommender systems in Technology Enhanced Learning.

On the other hand, the adoption of cloud computing has risen constantly during the last years [76]. This trend also is observed in the educational arena where research on cloud in learning technologies has increased in last few years. This increase of interest on cloud in learning technologies has enabled a shift from monolithic systems into flexible e-learning services [77] and the proliferation of service- oriented e-learning environments [78] where e.g. personalized educational services are given in a distributed way but not in a unique central Learning Management System. In this section we present some of the most recent works regarding the application of cloud technology into the learning domain.

Cloud technologies are a set of easily accessible and virtualized resources that can be dynamically adapted allowing an optimum resource utilization [79]. A cloud technology allows users access onto different services on Internet through diverse devices. It also provides service providers with several advantages, such as availability, integration of multiple services, flexibility and scalability [80]. This makes cloud technology an attractive option for deploying applications that require a high level of computational power. Hence cloud technology has taken into account to support services that offer educational resources to academic communities.

In this sense, in [81] Mikroyannidis states that the cloud offers a lot of services for building adaptive and customisable CLE. He also explains how CLEs extend the borders of the learning environments beyond of educational organization. Additionally this work proposes a learning scenario based on the use of cloud learning services. Thus, to take advantage of these features educational institutions are also moving towards providing their services by using cloud technologies. However, in the educational domain, services are scarcely adapted and offered in cloud.

Some distributed oriented learning architectures have been proposed such as OKI [82], KnowledgeTree [83], or in the ELENA project [84]. OKI (Open Knowledge Initiative) proposes an architecture based on layers that includes educational services that can be accessible invoking remotely a set of web services. OKI defines and provides the interfaces for these services, and can create final applications that offer a user interface, combining different invocations from these services. OKI is focused on educational services that can be found in a typical LMS such as assessment or authorization, and the invocation of the implementation of these interfaces can lead to build a specific LMS. However, OKI is not focused on typical adaptive e-learning services. On the other hand, KnowledgeTree [83] and ELENA [84] both represent distributed architectures for the personalization of resources for technology enhanced learning. Nevertheless, these architectures focus on generic aspects of adaptation of e-learning resources, but do not go into details of the specifics for affective learning. KnowledgeTree and ELENA would be complementary to our work as they can be used as general frameworks for distributed adaptive learning, while our architecture would bring the details for affective learning. In addition, OKI, KnowledgeTree and ELENA do not focus on analyzing important aspects of cloud computing as the distributed computation of a task, which is also addressed in this work.

Several approaches of cloud architectures promote improvements to services in the e-learning area by using cloud technologies. Thus, they try to overcome challenges faced by educational institutions. In [85] Masud and Huang propose an e-learning cloud architecture to allow the migration of e-learning systems from schools to a cloud computing infrastructure. They describe an e-learning cloud architecture made up of five layers: infrastructure, software resource, resource management, service and application. This proposal describes a general architecture for e-learning; nevertheless it does not focus on how to implement an e-learning service in a cloud architecture.

As CLE appears as a set of available tools on the internet that allows ubiquitous access to an academic community, it is evident that the existing of Personal Learning Environments in the cloud are in an early stage of its developing. Currently the CLE has dealt in offering an environment that allows learners and teachers easy access to different tools for producing and consuming academic content. A related work is also presented by Al-Zoube in [86], where he proposes a cloud computing based solution for building a virtual PLE. This proposal consists of allowing the learner access to different tools offered on internet such as iGoogle, Google docs, YouTube, etc. however such as Stein et al. state in [87] public clouds generally meet the common base of user requests, but they may not be designed to meet educational needs. In addition, today learners are demanding specialized learning services in order to improve the learning results. Hence, the educational domain requires design and deploy services in order to built a true educational Cloud.

Following the above approach, in [88] Madan et al. present a cloud-based learning service model. This proposal describes comprehensively the cloud computing services as a key aspect of cloud computing model. The authors focus on services and available models to be deployed into cloud architecture. They claim that institutions should use the existing cloud infrastructure offered by companies such Google, Amazon and others. Then educational institutions should focus on defining the cloud service layer to implement it into the cloud architecture.

There is a known necessity of building a CLE based on specialized services rather than the traditional tools found in PLEs. However as the necessities of resources and services for learners and teachers are variable, we must offer a service to adapt the PLE in the cloud according to these necessities. In other words, CLE needs a recommender system to fill this gap. Recommender systems have been extensively deployed, however few systems operate in the education arena [89]. Then in this work we propose a generic architecture for a system in the cloud to recommend learning elements according to the affective state of the learner. Thus learners will be able to adapt their PLE and CLE. Additionally we provide details of the architecture implementation of this specialized service.

Chapter 3

Detection of emotions in new learning environments

The inclusion of affective information can be achieved with some level of certainty through the analysis of the actions done by students in a learning environment [90].

There are different emotions related to a person described in the literature. Four of these emotions have been predominant in the literature of detection in learning environments. These four emotions are boredom, confusion, frustration, and happiness. Using the data available in the learning environments, detection of these four emotions is tried to be inferred. As shown by Baker et al. in [32], the observation of confusion and engagement has been proven to correlate positively with learning gains, while boredom and frustration present a negative correlation. Therefore, it is interesting to try to infer these emotions as they correlate with learning gains.

As presented in the related work chapter, there has been intensive work in the detection of emotions based on different types of sensors in different learning environments. In addition, there has been some works of detection of emotions in some learning environments such as Intelligent Tutoring Systems. But the detection of emotions in other learning environments needs to be explored. The selected environments for the detection of emotions are very different:

- A programming learning environment. In this environment, the students have to use and learn some tools by their interaction with them.
- A MOOC environment. In this environment, the students have to prove that they

learn something by watching videos, solving exercises, etc.

The proposed models are very different for these two learning environments. This difference is caused because the types of events are very different in their nature.

The programming learning environment use a probabilistic model that tries to manage the uncertainty of the prediction of the emotions based on different types of events on the tools that the students should learn. On the contrary, the MOOC environment defines a rule-based approach combined with linear regression models.

The difference in the selection of methods for these two learning environments is justified mainly because the second environment does not make use of a lot of events, because we only focus on some events related to solving exercises, while the first learning environment makes use of a lot of events as there are many different tools that can be used by the students. A probabilistic model can match better with a high number of events to manage the uncertainty, while a rule-based model might match better in a case with a more reduced number of events.

However, a probabilistic model might also be used for the MOOC environment or a rule-based model might be also used for the programming learning environment.

3.1 Detection in a programming learning environment

The first learning scenario focuses on an environment in which the students have to do a set of tasks directly with some tools that they should learn. Specifically, these tools are related to software development, and the students should learn how to use these tools in order to make programs. Therefore, the detection of emotions takes place during the process of learning to develop software. During the activity of software development, the learner can experience several affective states that affect her performance. For instance, being unable to find the origin of a compilation error or to understand an error message can generate the emotion of frustration.

3.1.1 Description of the learning environment and type of events

The learners are instructed to use a virtual machine containing every tool they will need during the process of learning programming. Specifically, the used programming language is C but the solution might be generalizable to other programming languages. The programs that students make during the term are submitted to the teaching staff through a versioning control system. The virtual machine is configured to collect the interaction of students with a specific set of tools. Some of these tools are specific for the subject domain (compiler, debugger, memory profiler, command line and versioning control system) while others are more generic (text editor and web browser). Fig. 3.1 presents a screen capture of the virtual machine environment.

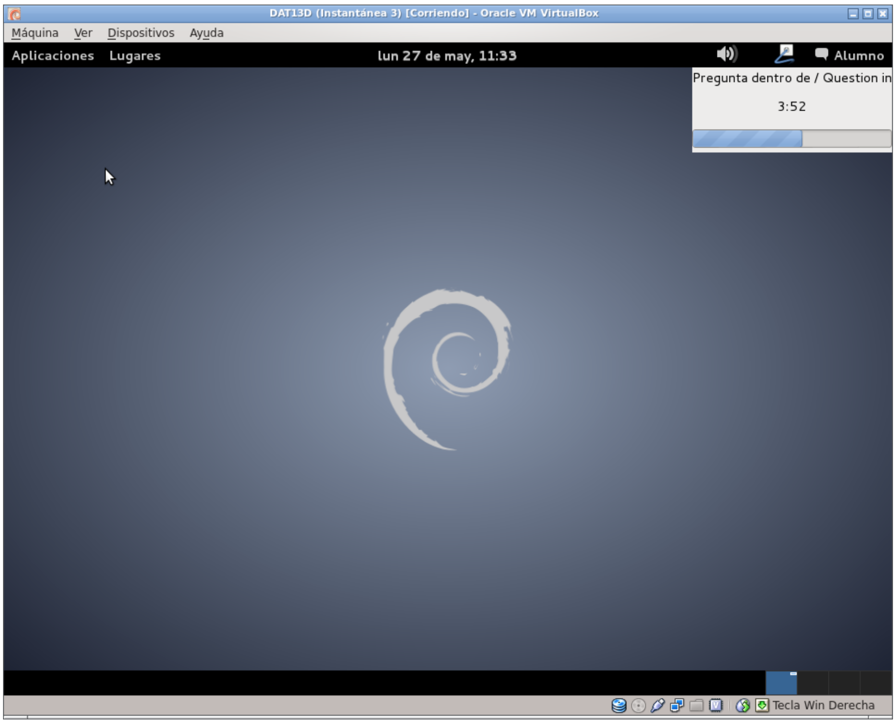

Figure 3.1: Screen capture of virtual machine with capturing tools.

The collection of events is based on previous work described by Pardo and Delgado Kloos in [91]. The interaction with the mentioned tools, along with the time and type of action performed by the learner, are stored in a file or log. The information stored in these logs depends on the data provided by the associated tool. For instance, a web browser provides the web address (URL) accessed by the learner and this is the only data stored in the web browser log. A compiler generates a detailed report in case the program compilation is erroneous; the virtual machine stores this report in the compiler log. Logs are stored as plain text files within hidden folders in the workspace of the learner.

During the enactment of sessions in the computer laboratory, students were instructed to send their work through a versioning control system. The logs of the used tools are sent along with the learner's work to a centralized server. This process is repeated each time that the learners submit their changes; thus, logs of collected events are stored in an incremental manner in the server. Gathered events can be analyzed at the end of a specific term (e.g.at the end of the academic year) or in a periodic way (e.g. every day or every week).

In addition to the logs created by the virtual machine, we also gather events from external sources. One of these sources, given our educational context, is the virtual learning environment or learning management system; which is an instance of Moodle. The events generated in Moodle include interactions of the learners with the course (e.g. accessing to the course page or the list of participants). Another source of events is the web server that hosts the course resources, from where we gather the accesses of learners to the web pages of the course.

In order to analyze the events stored in the central server, the logs must be normalized because the format of each file depends on the data obtained from the associated tool. The normalization is achieved through a lexical analysis of the files content. As a next step, the normalized information is stored in a generic format. In our scenario, we have chosen the format Contextualized Attention Metadata (CAM) [92], which allows to express user actions in a computerized environment. In [91], the reader will find further information of the process to capture events, and [93] presents details of another experiment that has relied on the same process.

3.1.2 Proposal for the detection of emotions using Hidden Markov Models

Previous studies have considered the relation between event patterns and the appraisal of emotions [42]. These models agree on the fact that events can be classified as favorable or adverse in the path to reach an objective. In the context of this work, the objective of the learner is to successfully finish a learning activity in the domain of software development.

Taking this objective as the starting point, we define a model to analyze a sequence of events and relate them to the emotion with the highest probability to generate such sequence. For instance, a sequence with most of its events adverse to the learner's objective (e.g. experiencing compilation or memory profile errors) would be related to the emotion of frustration.

Given the requirement to detect patterns in a sequence of observations, we have used Hidden Markov Models (HMM). An HMM has two sets of elements: an alphabet or set of symbols, and a set of states. The behavior of the model is defined by three sets of probabilities: a list of probabilities of each state to be the initial state, a matrix of production probabilities for each combination of state and observation, and a matrix of probabilities for transitions between states. Thus, one can compute the probability to compute an observed sequence of symbols using the forward-backward algorithm.

The symbols or observations to use in our proposal are built upon the events previously captured. A subset of symbols are obtained directly from the event generated in the virtual machine, like the symbol *command*. In other cases, the symbol resulting from an event depends on the information associated to such event (e.g. a compilation event can generate symbol *compile-error* or *compile-ok*). The inverse process can also occur, different events can generate the same symbol (e.g. all events generated in Moodle's forum generate the symbol *forum*). Table 3.1 presents a list of observations taken into account along with their descriptions and the tools that generate them either in the virtual machine or in an external system.

The proposed model defines five states for the HMM, described as follows:

Activity work (E_1). This state implies a fluid workflow in software development. Most of the compilations are successful and most of the work occurs within the programming environment. Erroneous compilations can occur in low amounts and get solved quickly. The learner can also communicate with the instructors and other learners through the forum, but should not review either subject nor external resources.

Challenge (E_2). In this state the learner gets saturated and cannot quickly solve a compilation or execution error. Interactions with the text editor and compiler are often, as used to solve the error. Given that this state represents the point where the learner finds the error, it is not expected to observe interactions with tools other than the compiler, memory profiler, and to a lesser extent, command line.

Information search (E_3). This state is differentiated by the access to support resources to look for the solution to the challenge found in E_2. Learners in this state tend to use the web browser to find information related to the challenge. A high level of interaction with Moodle forums is also expected. In addition, the learner

Table 3.1: Symbols observed in model.

Observation	Description	Emitting tool
command	Instruction entered in command line	Bash
forum	Any interaction with the subject forum: accessing a thread, create a new thread and sending a message	Forums in LMS Moodle
compile-error	Compilation with erroneous outcome	GCC, Java
compile-ok	Compilation with successful outcome	GCC, Java
debugger	Interaction with debugging tool	GDB
ide	Interaction with development environment	KDevelop
lms	Interaction with tools in the LMS other than the forum	Moodle
text-editor	Starting or stopping usage of text editor	Kate
resource-external	Accessing material unrelated to the subject	Google Chrome, Mozilla Firefox
resource-internal	Accessing resources related with the subject (assessed by checking URL)	Google Chrome, Mozilla Firefox
resource-search	Access Google search website	Google Chrome, Mozilla Firefox
memory-ok	Debugging memory management with successful outcome	Valgrind
memory-error	Debugging memory management with erroneous outcome	Valgrind

accesses to local documents and documentation of the programming environment. Interactions with software development tools are not expected.

Solving challenge (E_4). In this state the learner finds and applies the solution to challenge described in state E_2. There is a low number of successful compilations and none erroneous. This state doesn't generate any symbol related to searching information in forums or in class material.

Distraction (E_5). Learner is not working on the software development tool but on an unrelated activity. Among other events, we can observe accesses to websites unrelated to software development. No interactions with software development tools are expected.

Table 3.2: Initial probabilities for each state and symbol.

Symbol	E_1	E_2	E_3	E_4	E_5
command	0.25	0.05	0.00	0.05	0.00
forum	0.10	0.00	0.30	0.00	0.00
compile-error	0.15	0.70	0.00	0.00	0.00
compile-ok	0.10	0.00	0.00	0.50	0.00
debugger	0.05	0.00	0.00	0.05	0.00
ide	0.05	0.00	0.00	0.05	0.00
lms	0.10	0.00	0.10	0.00	0.05
text-editor	0.10	0.00	0.00	0.00	0.00
resource-external	0.00	0.00	0.10	0.00	0.95
resource-internal	0.00	0.00	0.30	0.00	0.00
resource-search	0.00	0.00	0.20	0.00	0.00
memory-ok	0.05	0.00	0.00	0.35	0.00
memory-error	0.05	0.25	0.00	0.00	0.00

Each of the states has a probability to emit each symbol previously described. The probability assignation depends on the description of the state and the tools related to it. Table 3.2 presents the probabilities initially assigned in our model. These probabilities can be obtained by training the model with Baum-Welch algorithm.

While the probabilities of observation emissions are constant for each emotion, transition probabilities change according to the learner's affective state. We focus on four affective states commonly observed in learning experiences: happiness, frustration, confusion and boredom. Even though the last two are usually classified as cognitive states, a high level of correlation has been observed between them and learning gains [94]. Furthermore, confusion and boredom can be related with the behavior of a software developer.

For instance, in this scenario a confused learner works following a trial-and-error approach. A bored learner tends to do unrelated tasks, as well as browsing web sites with no relation to the class content.

For the case of happiness, we expect fluid work from the learner. This implies that most of the transitions have the target state *Activity work (E_1)* disregarding the origin state. A learner can find a challenge but should find its solution with high probability. There must be a minimum amount of events from state *Distraction (E_5)* because the learner is motivated on the development task. Table 3.3 shows an initial proposal for the transition probabilities of the happiness affective state.

Table 3.3: Transition probabilities for affective state of happiness.

State	E_1	E_2	E_3	E_4	E_5
E_1	0.60	0.10	0.10	0.10	0.10
E_2	0.30	0.20	0.30	0.10	0.10
E_3	0.40	0.10	0.20	0.20	0.10
E_4	0.60	0.10	0.10	0.10	0.10
E_5	0.60	0.10	0.10	0.10	0.10

Table 3.4: Transition probabilities for affective state of frustration.

State	E_1	E_2	E_3	E_4	E_5
E_1	0.40	0.30	0.10	0.10	0.10
E_2	0.10	0.50	0.20	0.05	0.15
E_3	0.25	0.50	0.10	0.05	0.10
E_4	0.70	0.15	0.05	0.05	0.05
E_5	0.30	0.30	0.15	0.05	0.20

Transition probabilities for the frustration model are shown in table 3.4. The main characteristic in this case is the high probability to transition into state *Challenge (E_2)*. State *Information search (E_3)* is expected at a low rate due to the learner frustration. There is a moderate probability of staying in state *Distraction (E_5)*.

There is a similarity between the transition probabilities for frustration with the ones for confusion because of the resemblance of their causes in a software development environment. The main difference in the definition of their HMMs is that when confused, a learner has more probability to stay in state *Challenge (E_2)*. This model also defines a low occurrence of transitions from *Information search (E_3)* to *Solving challenge (E_4)*; while the probability of transition into *Distraction (E_5)* increases and there is a high probability of staying in such state. Transition probabilities for confusion are shown in table 3.5.

Table 3.5: Transition probabilities for affective state of confusion.

State	E_1	E_2	E_3	E_4	E_5
E_1	0.40	0.30	0.05	0.05	0.20
E_2	0.15	0.50	0.15	0.05	0.15
E_3	0.20	0.30	0.20	0.10	0.20
E_4	0.60	0.20	0.05	0.05	0.10
E_5	0.20	0.30	0.05	0.05	0.40

Table 3.6: Transition probabilities for affective state of boredom.

State	E_1	E_2	E_3	E_4	E_5
E_1	0.40	0.10	0.10	0.10	0.30
E_2	0.10	0.20	0.20	0.10	0.40
E_3	0.15	0.20	0.20	0.15	0.30
E_4	0.40	0.10	0.05	0.05	0.40
E_5	0.20	0.10	0.05	0.05	0.60

Finally, table 3.6 shows the transition probabilities for the boredom model. In this case, the learner is expected to switch from a software development tool or looking for information to doing an unrelated task. The main characteristic of this model is the high probability of transitioning and staying into state *Distraction (E_5)*.

The process to identify the current affective state of the learner consists of computing the probability for a event sequence to be generated by the model of each state. First stage of this process requires the definition of a sequence length, l. At this stage, events must be already converted into their corresponding symbols. The next step is to compute the generation probability for the event sequence with all models; this step applies the forward-backward algorithm. Resulting probabilities will present a low order of magnitude because it corresponds to the probability of generating an exact sequence of length l. The affective state with the highest probability is selected as the one being experienced by the learner.

The process of detecting affective states can be understood as a transition between levels of information. Figure 3.2 is a graphical representation of processing events into high-level actions related to predefined states. The top level is composed by events captured directly at the virtual machine, the learning management system and the subject web server. The middle level contains symbols built upon events. These symbols have the appropriate level of information to be related with software development activities, included in the bottom level. The bottom level represents the affective state of the learner, obtained from the transition of events from the first layer.

3.1.3 Implementation of proposed models and application example

The event database is stored in the system MySQL. The translation of events and process modeling described above have been implemented on Python. The implementation uses

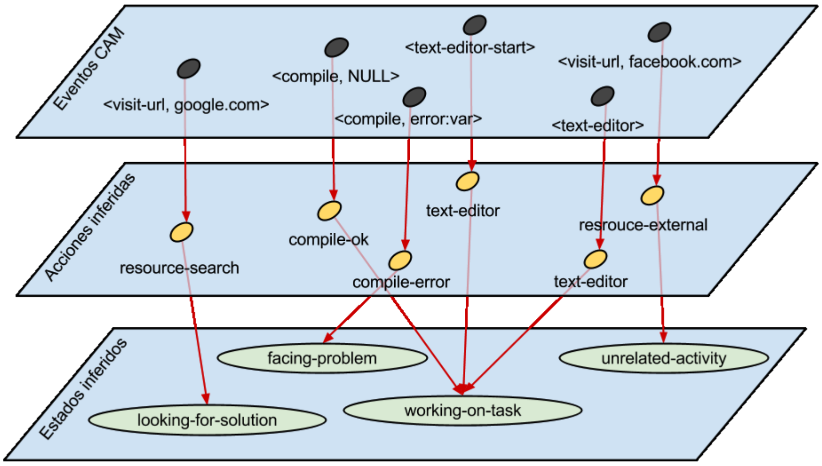

Figure 3.2: Representation of conceptual hierarchy of learning events.

the library GHMM[1] to treat HMM, which includes forward-backward and Baum-Welch algorithms. Results of the process as stored in plain text files to be analyzed in a statistical evaluation stage.

An example of the execution of the model is presented with the goal of completing and enabling its understanding. For this example we have defined a sequence length of 20; meaning that at least 20 events must be captured in order to start the detection of the affective state. Table 3.7 shows a sequence of the symbols generated from the events captured in a real learning activity. Respective columns for each emotion correspond to the logarithm of the probability for the event sequence to be produced by the HMM of the affective state. Thus,

$$log(P(S_t|H_E))$$

where S_t is the sequence of captured symbols at time t and H_E the respective HMM for emotion E. The logarithm function allows simplifying the presentation of such low probability values. As expected from the probability magnitudes, all resulting values are negative. The table highlights the affective state with the highest value, which represents the highest probability. We can observe how initially the learner generates a sequence that seems normal at the moment, starting by using the command line and then accessing two resources related to the subject. With these initial states, the probability for the

[1] http://ghmm.org/

student to experience happiness is the highest and this is selected as the current affective state. Then, the learner generates three events of erroneous compilation, followed by two events of searching a resource and an access to an external resource. By observing the results, we can appreciate a decrement of the probability at the happiness column up to row 9 where the value is considerably lower than at row 1. Meanwhile, the probabilities of the other three affective states increase and confusion shows the highest value. This behavior continues as the learner searches for a solution to the found challenge as can be seen at rows 10 to 14. At row 15, there is a successful compilation event and the trend of probability values changes. By row 20 there have been three successful compilations and the affective state of happiness is again the one with the highest probability. This is sustained until the end of the example, presenting a sequence of events appropriate for the process of learning software development: compilations and memory analysis mostly successful and use of text editor and debugger.

3.1.4 Discussion and limitations

This HMM based model of detection of emotions in a software environment has been implemented successfully. The fact that just the transition probability matrix among states is required for detecting each emotion makes easier the implementation. Moreover, it simplifies the extension of the model to new affective or cognitive states that might be of interest in a software development environment.

The main weakness of the proposed model is the high number of variables that influences the detection of affective states. The probabilities of generation of symbols are related to 5 elements and 13 symbols, so there are a total of 60 possibilities for its definition. In addition, the transition probabilities among the 5 states give other 20 possibilities. We have tried to reduce this weakness with a coherent definition of the initial probabilities, as explained before. In addition, if there are enough data, these probabilities might be adjusted in order to improve the predictions of the model.

Another weakness caused for the use of HMM is that context information cannot be included during the detection process. An example is information related with the student location in order to analyze whether doing the software development activity at home or at the university might affect the students' emotions. Other type of information is related to time in order to analyze if some emotions are more frequent at specific times or e.g. during the weekend. As these type of information cannot be included directly in

Table 3.7: Example of symbols generated and the logarithm of the resulting probability for each emotion.

No.	Symbol	Happiness	Frustration	Confusion	Boredom
1	command	**-53.75**	-62.49	-58.41	-57.30
2	resource-internal	**-54.56**	-62.80	-59.28	-57.86
3	resource-internal	**-54.75**	-63.35	-59.09	-57.82
4	compile-error	**-54.12**	-61.33	-57.41	-56.69
5	compile-error	**-53.02**	-59.23	-55.09	-55.39
6	compile-error	**-52.46**	-57.50	-53.49	-54.68
7	resource-search	**-53.30**	-58.13	-54.52	-55.48
8	resource-search	**-53.74**	-58.92	-54.49	-55.55
9	resource-external	-53.70	-58.57	**-53.54**	-54.28
10	compile-error	-52.92	-56.92	**-51.90**	-53.65
11	resource-search	-53.38	-57.17	**-52.22**	-53.96
12	resource-search	-53.25	-57.49	**-51.85**	-53.76
13	compile-error	-52.48	-54.93	**-50.46**	-52.76
14	compile-error	-51.42	-52.46	**-48.67**	-51.83
15	compile-ok	-50.99	-52.22	**-49.03**	-51.77
16	debugger	-51.53	-52.11	**-49.78**	-52.72
17	debugger	-52.07	-52.24	**-50.72**	-53.61
18	compile-ok	-52.47	-51.13	**-50.61**	-53.26
19	compile-error	-49.80	-49.23	**-49.11**	-52.63
20	compile-ok	**-50.38**	-51.22	-51.19	-53.36
21	debugger	**-51.93**	-53.63	-53.80	-55.23
22	memory-ok	**-53.17**	-54.67	-54.88	-56.32
23	memory-error	**-52.85**	-53.27	-54.09	-56.20
24	compile-ok	**-51.36**	-52.27	-53.92	-55.94
25	memory-error	**-52.29**	-53.68	-55.19	-56.96
26	text-editor	**-53.87**	-55.68	-56.81	-58.85
27	compile-ok	**-52.85**	-54.18	-56.32	-58.04
28	text-editor	**-52.17**	-54.66	-56.50	-57.99

HMM, a possible solution would be to make the experiments in each one of the selected environments and compare the obtained results.

3.2 Detection in a MOOC learning environment

The second learning scenario is a MOOC, in which the students have to do a set of learning activities such as watching videos or solve exercises. The learning activities are about the concepts and processes they have to learn but they do not use directly the tools they have to learn as a difference with the previous scenario. For simplicity, we only focus on the students interactions with the exercises.

The field of educational technology has been disrupted by the emergence of MOOCs. On one hand, students are continuously enrolling in order to increment their knowledge about certain topics. This high level of demand is being compensated by educational institutions and professors that are producing and delivering more courses, either because of the uncertainty of MOOCs impact in society or for altruism [95]. In a similar way, new MOOC platforms are being launched, and those that were already established are gaining visibility at accelerated pase (e.g. Coursera, edX, FutureLearn or MiriadaX).

The high expectation created by the appearance of MOOCs has provoked a high-paced evolution of the pedagogical and technical approaches in order to improve their effectiveness. One of the lines of research with this goal is to leverage the knowledge gained from adaptive learning [96]. Educational tools like intelligent tutoring systems and recommender systems have demonstrated the benefits of personalizing the educational resources and instructional design to the profile of the learner. The application of adaptation becomes more evident when participants in a MOOC can be categorized according to their engagement [97]. For instance, some researchers are also proposing the generation of personalized action plans within MOOC platforms in order to increase the learners' engagement [98].

In an adaptive educational system, the learner profile can include information like current learning skills, learning style, learning goals and accessibility needs. Along with these characteristics, students' emotions can enrich the contextual information available in the adaptive system. Furthermore, previous studies have shown the co-occurrence of some emotions with learning gains [32]. Thus, the inclusion of affective information in MOOCs can be used to customize the learning experience or to suggest the teaching staff to perform an intervention to improve the outcomes of the course.

3.2.1 Description of the learning environment and type of events

The specific learning environment where the detection was done is the Khan Academy platform. Khan Academy was one of the first systems to provide a set of educational resources similar to those included in a MOOC [99]. There are some differences in the pedagogical approach followed in Khan Academy with respect to other MOOCs because of the lack of the concept of a course. Learners cannot enroll in a course to follow a specific sequence of activities, and it is not possible to create a community around each topic that could encourage the creation of a learning community. However, two elements that are used in this system in a similar way than in MOOCs: videos and exercises that provide immediate feedback.

The Khan Academy platform, which was previously available as open source under the MIT license, presents educational content in video format hosted in YouTube. Following the approach seen in MOOCs, the videos tend to be short and they are assigned to a specific topic to be learned. The platform keeps track of the videos wathed by the student in order to suggest the ones to watch next. Another element common in both MOOC platforms and the Khan Academy platform are exercises. These consist of multiple-choice questions about the concepts explained in the videos. The exercise can include mathematical expressions that improve the readability of the question posed. If the learner is not completely sure about the answer to the question, the tool includes the option to provide hints. A screen capture of an algebra exercise is shown at Fig. 3.3 as an example of the exercise element.

In addition, the Khan Academy platform provides mechanisms for gamification which are not common in other platforms. For instance, the platform attributes badges to a learner that has achieved of a specific goal, such as watching a given number of videos or answering some exercises correctly.

3.2.2 Proposal for the detection of emotions using rule-based models

For the selected four emotions, we have selected only those events occurring during the last hour. This decision relies on the fact that more recent events tend to affect in a higher level the emotional state of a person in any context. Furthermore, those events occurring within the hour have a different weight on the current emotional state, depending on how recent they are. For instance, an event occurring one minute ago has a greater weight than an event occurring 50 minutes ago. Studies in the field have

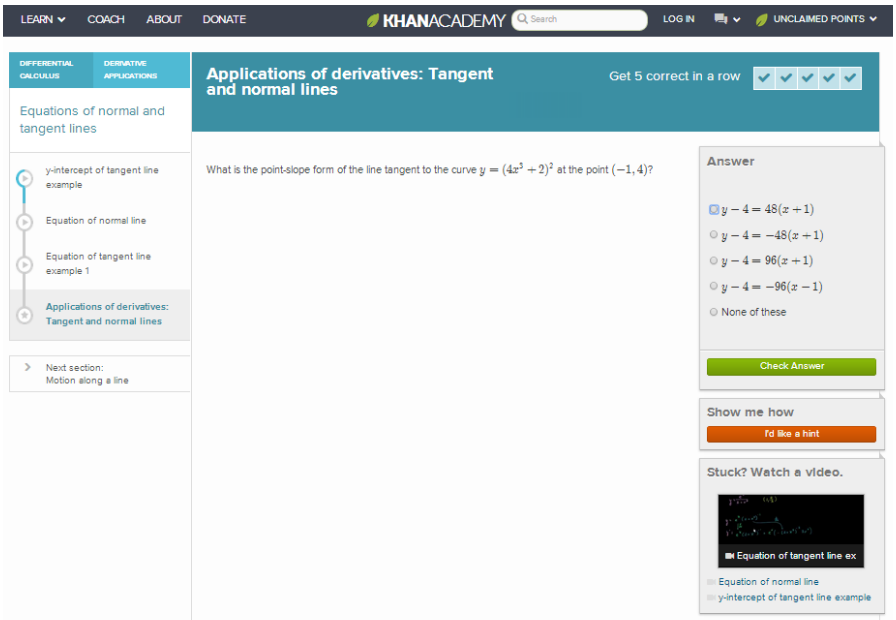

Figure 3.3: Screen capture of an exercise interface in the Khan Academy Platform

used similar criteria in order to define the duration of sessions in which the emotions are inferred [100, 41].

Detection of frustration

The inference of the frustration of the student (SF) is based on the learner's attempts of the exercises. Those exercises are considered not to increase the frustration of the learner. On the other hand, if the last try of the student has not been correct, there is understood to be a level of frustration, with a behaviour characterized by the following equation, where A is a constant to be defined.

$$f(t) = \begin{cases} 0.7, & t < 20 \\ A * (t - 20) + 0.7, & 20 \leq t < \frac{0.3}{A} + 20 \\ 1, & t \geq \frac{0.3}{A} + 20 \end{cases} \tag{3.1}$$

If the student has not tried to solve the exercise, the frustration is understood to be null at the beginning and to increase with the pass of the time. The following equation is proposed to calculate the frustration generated in this case. As in the previous case, B is a constant to be defined according to the learner.

$$f(t) = \begin{cases} 0, & t < 20 \\ B * (t - 20), & 20 \leq t < \frac{1}{B} + 20 \\ 1, & t \geq \frac{1}{B} + 20 \end{cases} \tag{3.2}$$

As mentioned above, the frustration generated by the exercise is weighted according to its time of occurrence. The weight of the exercise is calculated with the following equation, where E represents the exercise and M the set of minutes in which the exercise occurred.

$$w(E) = \frac{1}{73810} \sum_{i}^{M} (60 - i)^2 \tag{3.3}$$

The same equation is used to calculate the exercise weight for the other emotions. Finally, the emotion of the student is incremented by the level generated by the exercised weighted in equation 3. This calculation is used in every emotion model given the incremental change provoked by each exercise. Thus, the equation used to update the student emotion is the following, being SE the student's emotion, EE the increment of emotion generated by the exercise, and EW the weight of the exercise.

$$SE = SE + EE * EW \tag{3.4}$$

The result obtained is used as an index between 0 and 1 that indicates the level of frustration generated by the exercises in the platform. A diagram of the process to calculate the frustration following the equations described above is presented in Figure 3.4.

Detection of confusion

The process to infer the emotion of confusion is similar to the one used to infer frustration. Indeed, previous works have found difficulties to define models for confusion and frustration that do not correlate between themselves [101].

In our proposal, the logic behind the definition of this model relies on one of two events. The first is the case when the student is taking a long time to solve an exercise, similar to the detection of frustration but with a different slope. The second case consists in those situations where a learner has previously solved an exercise and in a later try

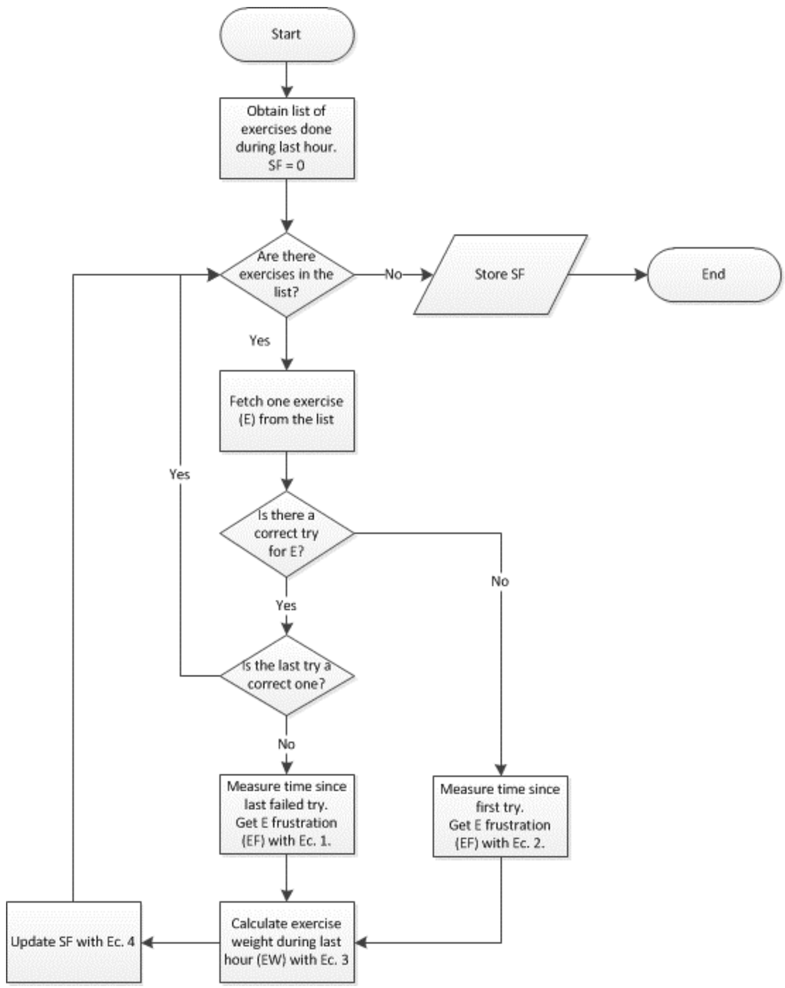

Figure 3.4: Flow chart of process used to detect frustration

the response is incorrect. Thus, the concept evaluated by the exercise is not completely clear for the learner.

The surrounding steps of the model coincide with those of the frustration model: each one of the exercises done by the learner during the last hour is analyzed in order to get its individual effect on the learner confusion. If a learner has given a wrong answer to an exercise that had been solved previously, it is understood that the confusion associated to that exercise is total (coefficient of 1). The equation used to calculate the confusion generated by the exercise (EC) is the following.

$$f(t) = \begin{cases} 0, & t < 5 \\ C * (t-5), & 5 \leq t < \frac{1}{C} + 5 \\ 1, & t \geq \frac{1}{C} + 5 \end{cases} \tag{3.5}$$

As in the previous cases, C is a constant assigned to each learner. It can be seen that the equation that describes the confusion in terms of elapsed time trying to solve an exercise follows the same structure that the one describing the frustration of the learner. The main change between the two models is the smaller offset on the X axis for the confusion model. The reasoning behind this decision is that the learner can be confused at a very early stage of the assigned task, while frustration is most common at a later stage, when the learner would have spent more time to try to solve the problem. The flowchart describing the process for inferring confusion is presented in Figure 3.5.

Detection of boredom

Our proposal for the inference of boredom relies on the flow theory proposed by [Csikszentmihalyi, 1997]. In this theory, a learner is understood to be bored when the difficulty of the challenges presented is lower than the recommended for her level of skills. On the other hand, learners whose skill levels are not enough for the problems to solve are understood to undergo through the emotion of anxiety. The process to infer the level of boredom of the learner follows the same initial steps described in the previous two models. The list of exercises responded during the last hour are analysed individually to calculate their individual effect and calculate a weighted sum of the complete set.

The main difference in this model when compared with those for frustration and confusion is the lack of a linear function to describe the level of boredom with respect to the elapsed time. In this case the calculation has been simplified by, first,calculating

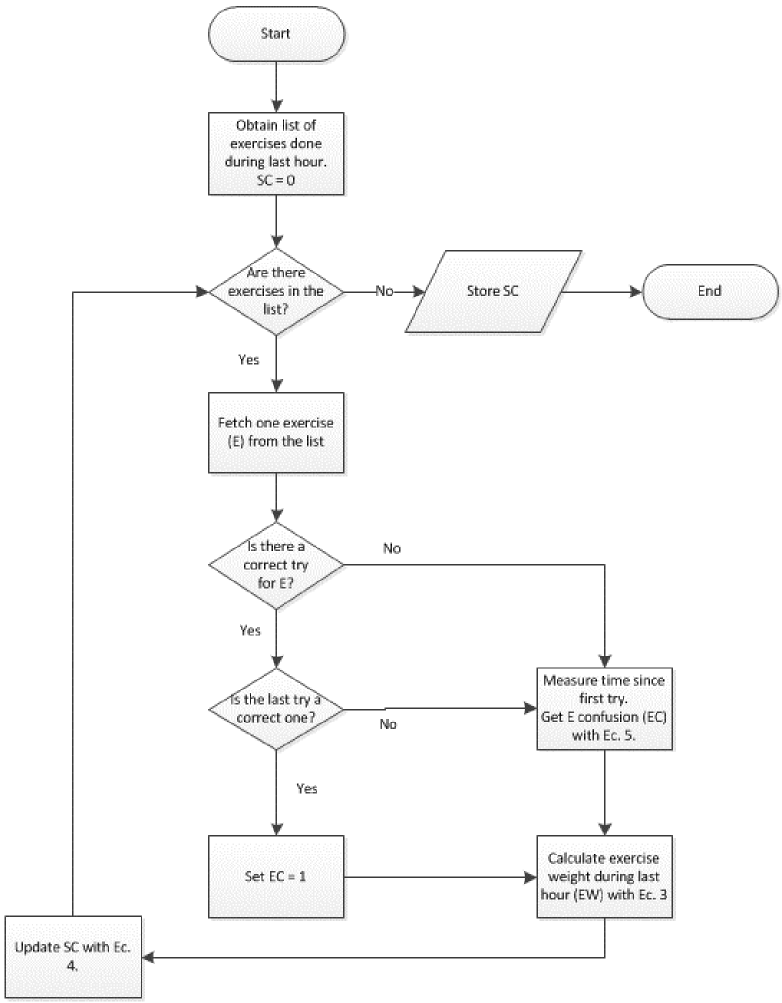

Figure 3.5: Diagram flow of model for the detection of confusion

the arithmetic mean and the standard deviation of the durations to solve exercises by the student. Then, we define that a problem assigned to a student is less than the expected if it is less than the mean minus one arithmetic mean. Another difference between previous models and this is that in this case an exercise is qualified to cause boredom to the student in a discrete way, this is, an exercise either cause boredom on the student or not cause it at all, while confusion and frustration were understood to be continuous functions. The diagram of the boredom model is illustrated in Figure 3.6.

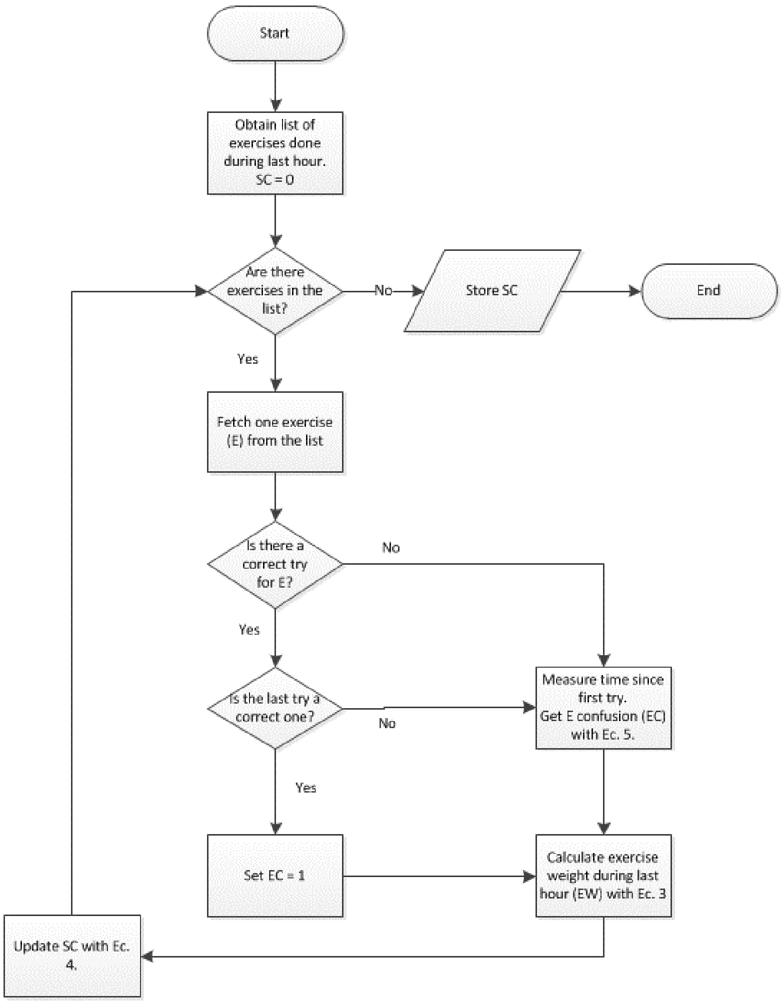

Figure 3.6: Flow of rules for the detection of boredom

Detection of frustration

The final model aims to infer the level of happiness that the student is experiencing as a result of the interactions with Khan Academy. Although this model maintains the idea of analysing only the events that occurred during the last hour, it also incorporates the analysis of gamification elements because of their direct relation with the happiness of the student. Specifically, we take into account the badges obtained by the learner because of solving each exercise. This hypothesis is based on Roseman's theory of discrete emotions [102] which states that an individual experiences joy when a rewarding event is certain to happen due to a circumstance.

As a point of reference, if the work done to solve the exercise does not provide a badge to the student, the happiness generated by that specific exercise is understood to be none. This model also includes the analysis of the emotion with respect to the time that the student has taken to solve the problem. Unlike the models for frustration and confusion, the level of happiness is understood to decrease with time. The process for this model has been illustrated in Figure 3.7.

3.2.3 Implementation of the models

The four models have been implemented and integrated into ALAS-KA, a learning analytics module for the Khan Academy platform [103]. This module extends the analytics capabilities provided by the platform and includes new metrics and visualizations of information obtained from the analysis of patterns of activities performed by its users. Furthermore, the developed application can be divided in two areas: the core inference engine and the presentation layer. The implementation of the inference engine follows a modular approach, having as a basic element the code in charge of the operations frequently used in the inference of emotions. Each emotion is implemented from this starting point. Thus, the implementation of other models for the detection of these and other emotions can easily be performed by the reutilization of these tools.

All of the implemented models take into account the ProblemLog provided by the Khan Academy. This log includes all of the information about users' interaction with exercises and the timestamp in which they have occurred. The process in charge of the analysis was scheduled recurrently in order that data processes could be executed as frequent as possible. The main advantage presented by this log is its role of gathering all of the data related to the actions of the student within the platform. This approach

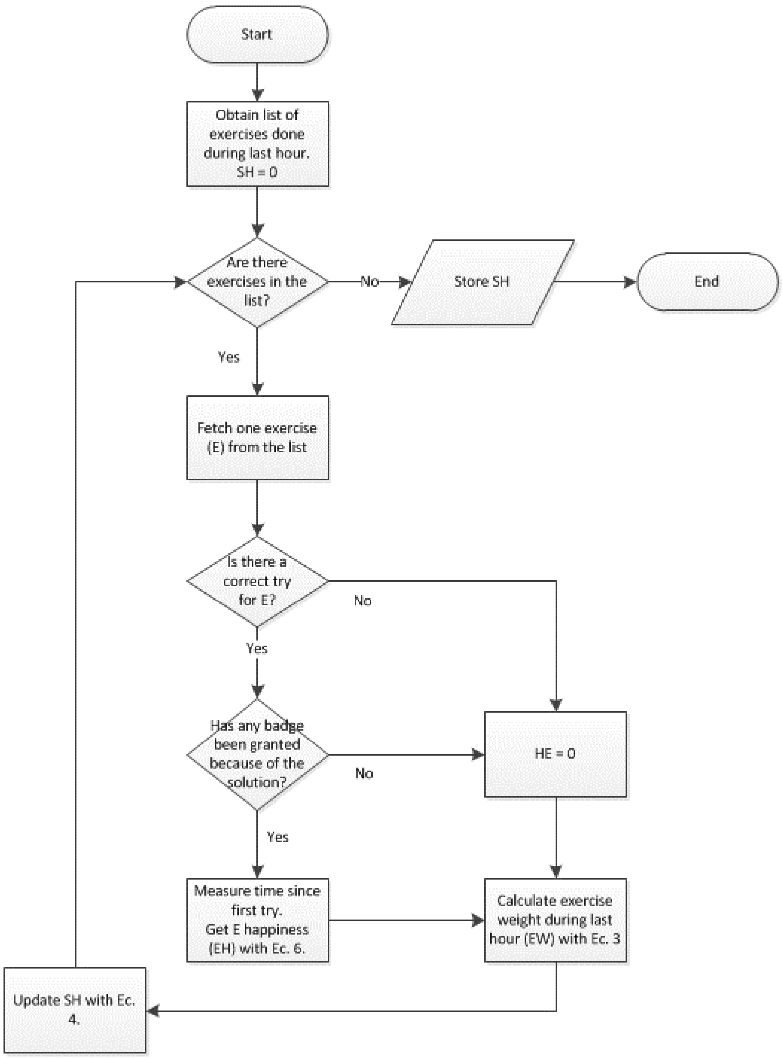

Figure 3.7: Flow for the detection of happiness

facilitates the process of collecting and normalizing the information generated.

The presentation layer includes the elements needed in the paradigm of ModelView-Controller (MVC). In this scenario, the view is a set of HTML templates that create a complete web page by using the data provided by the controllers. The final visualization presents a combo box to select the student whose emotions wanted to be inferred, and a table with the information of emotion changes in a timeline.

Chapter 4

Visualizations of emotions in learning environments

In this chapter we present several visualizations with the main objective of reflecting the emotional state of the learners, as individuals and as a group. These visualizations have been grouped in categories according to the main dimension used to show the information. Dimensions in this type of scenarios include time, learners, emotions, activities and context. This categorization enables the creation of a link between the goal that the visualization musth accomplish. For instance, when an instructor wonders about the overall effect of an specific activity on the learners, visualizations in both change and aggregated categories can be used as an initial approach.

The description of the four categories, their visualizations, and the discussion of their accomplishments and drawbacks are included as follows.

4.1 Time-based visualizations

The timeline visualization (Fig. 4.1) presents the fluctuations of each emotion for a given learner whose identity has been kept private. The X-axis represents the timeline, starting from date when the learner generated her first event, until the date when the learner generated her last event. The Y-axis represents the level detected for the emotion. As explained previously, the emotion level in our model goes from zero to one.

In the example figure, the learner had a peek of confusion and frustration by late September. In addition, the instructor can observe that by early November there was

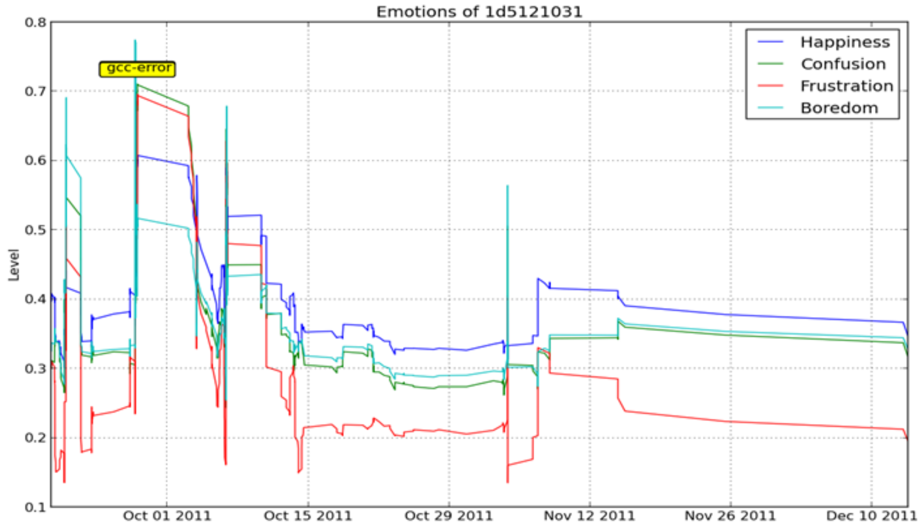

Figure 4.1: Timeline visualization, showing the level of emotions on the Y-axis and the day of the measure on the X-axis.

a peek of boredom detected in the learner; and the emotions kept in a controlled range during the rest of the term.

In addition, the visualization annotates interesting points that fall outside of a threshold, as well as the learning event that is related to that emotional state. The method used to identify the outstanding points is based on selecting those points higher than the mean plus a standard deviation. In the example, the highest peak occurs right after the learners had a problem to compile her program.

Thus, an instructor is able to identify when a learner is experiencing an extraordinary emotion. The instructor is also able to identify the event that caused the change of emotion in the model. This data combined give the instructor a considerable amount of intervention to act into the learning activity and assist the learner.

It is possible to superimpose the evolution of all students' emotions over time in the plot (as in figure 4.1 but including all students). Furthermore, the mean for the emotions of the classroom can be included. This is very useful to detect e.g. students with problems because are under the mean most of the time, students with punctual problems in the time, or periods where the emotion of most students are weak. In the

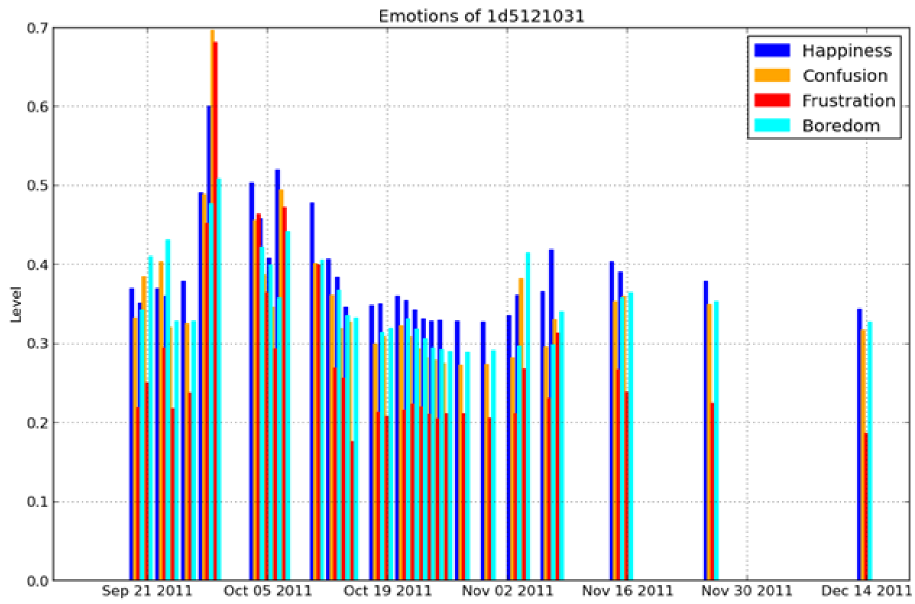

Figure 4.2: Daily amount of events color-coded with the affective state they reflected.

latter case, instructors can think in reviewing the learning activities during this period, as they produced a specific emotion for most students.

The visualization of daily accumulated events by their associated emotion is our other proposal of a time-based visualization. As seen in Figure 4.2, the visualization displays two aspects that are of interest for the instructor: the daily activity of the learner, deduced from the amount of events generated that day, and the emotion that was associated to each of those events. In the example, the learner was very active during the last week of October, but her activity was practically null the first two weeks of December. The color of confusion is not green as in the rest of the visualizations in order to distinguish it better from the boredom bar.

A combination of the previous visualizations results in a timeline that presents the evolution of time dedication of the student during the course and the average time dedication of the whole class. The visualization represents the accumulated time dedication of students: when the student selects a point of time in the horizontal axis, then the values of the vertical axis will indicate the accumulated levels of time dedicated until

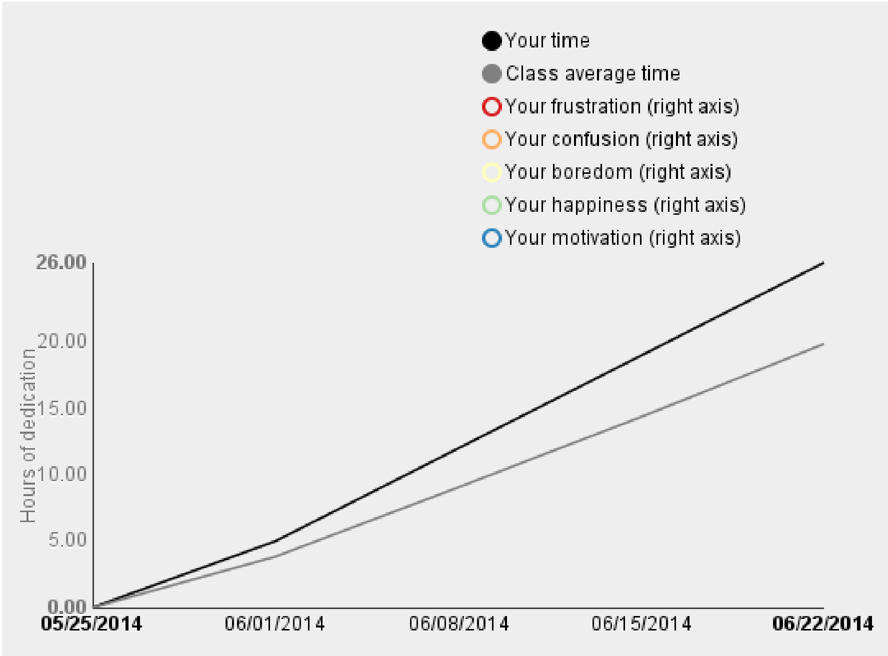

Figure 4.3: Visualization of the accumulated amount of time dedication and frequency of affective dimension. Students can also see the time dedication average of the class.

that moment. In addition, the timeline shows the evolution of each emotion during the course. Fig. 4.3 presents a capture of this visualization. An additional characteristic of this visualization is its interactivity, which allows user to show and hide a series of data in order to ease its understanding.

The next visualization in this category is a heat-map in which columns represent time units (e.g., days, weeks, months) and rows represent students. Each affective dimension is represented by a cell, while the frequency level of each emotion is represented through the intensity of the cell color (more intensity represents higher levels of this emotion). A portion of this visualization is shown in Fig. 4.4.

The final visualization in this category is a scatter-plot. Each affective dimension has a different scatter-plot associated to it. The X-axis corresponds to the exact date and time when the emotion takes place and the Y-axis presents the frequency value of the emotion associated to the scatter plot. Bubble sizes represent the amount of work dedication indicated in the given submission, and bubble colors indicate whether it belongs to the viewer or to another learner. Fig. 4.5 presents an example scatter-plot

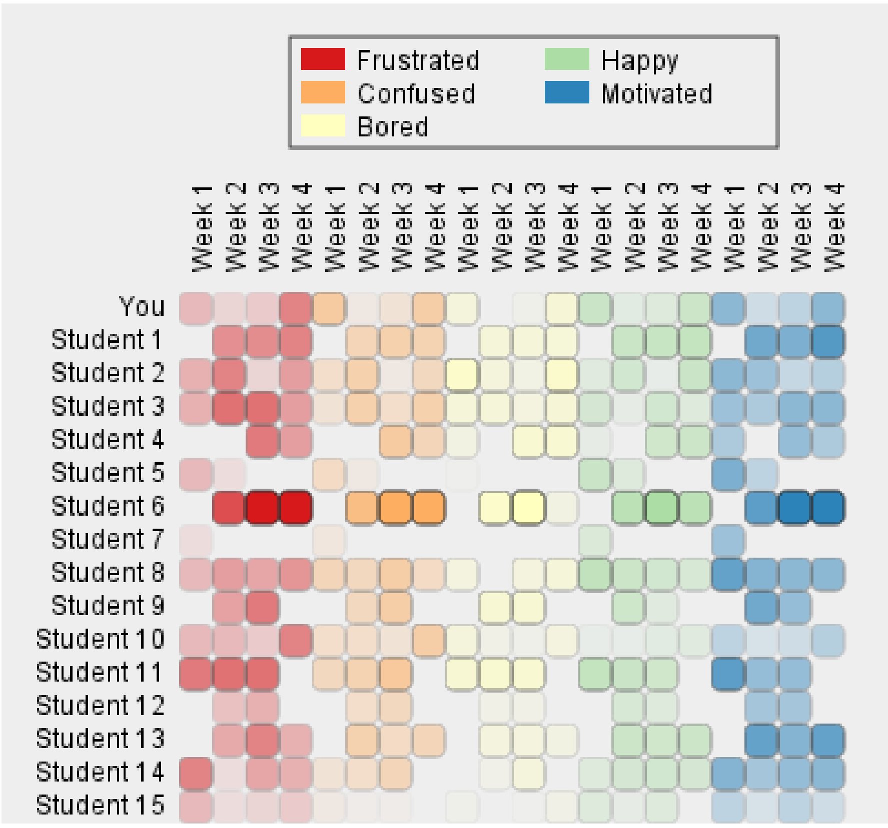

Figure 4.4: Heat-map of emotion frequency for each learner and each week.

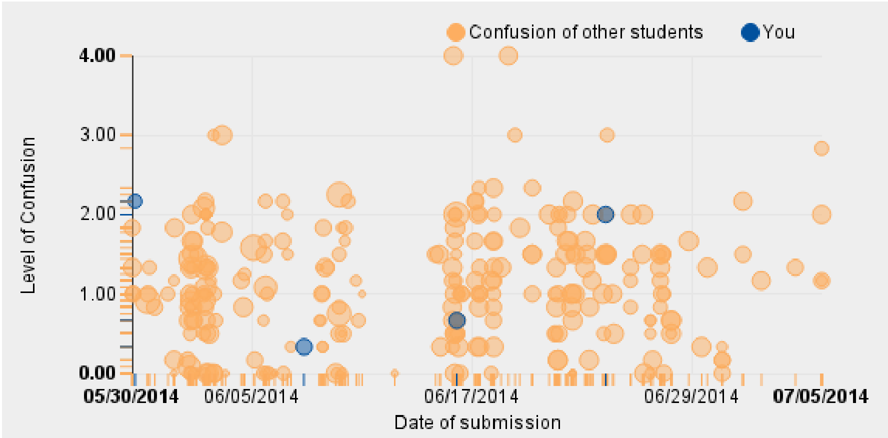

Figure 4.5: One of the scatter-plot showing the relation between the emotion frequency and work dedication along time.

for "confusion".

4.2 Context-based visualizations

The objective of this set of visualizations is to reflect the learning context in which the emotions were detected. This can help to understand the learning causes of some emotions. The first proposal in this set is the visualization of emotions by tool and type of emotion (Figure 4.6). This visualization shows the occurrences of events generated by a specific learner when expressing a given emotion and using a specific tool. With this type of graphic, instructors can know which tools generated better affective states to their students. In addition, comparing this type of graph with the mean of the classroom can lead to introduce clusters of students depending on the different effects generated by different tools.

In the example, the learner on the left felt confused, frustrated and even got bored while using the memory profiler (Valgrind, third from left to right). The compiler (GCC, first on the left) shows a similar level of confusion and frustration but these are overcome by the large amount of events associated to happiness. It is also worth noticing that the level of boredom while using the text editor (Kate, second from left to right) is relatively high compared with other emotions; this likely means that the learner constantly interrupted her programming task by accessing web content unrelated to the class.

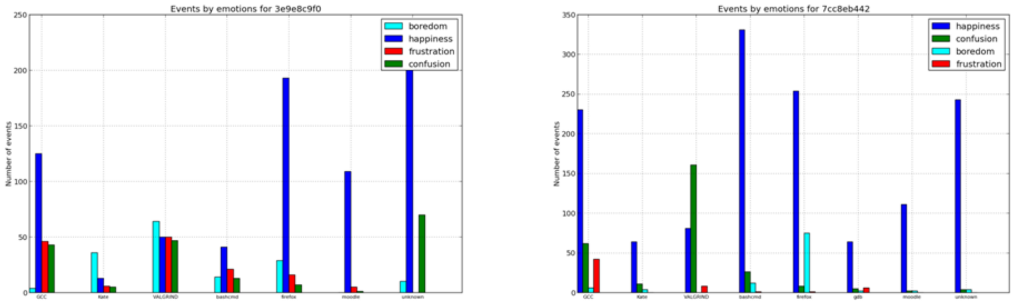

Figure 4.6: Visualization taking as context the application causing the affective state.

As a point of comparison, the learner on the right generated many events associated to confusion while using the memory profiler. The amount of events is even more than twice of those of the compiler. This indicates that the learner is having issues in particular with the memory profiler tool, and the instructor could reinforce topics related to the topic of memory management.

Another concept proposed as a context for emotions is the final score that the learners obtained in the course. The emotion, score and activity visualizations (Figure 4.7) display the relation between learners' emotions, the amount of events they generated in the learning scenario, and their final score in the course.

Unlike the previous visualizations, the score-related visualizations are not created upon the dataset of only one learner but upon the whole class group. This allows a direct comparison of the scores obtained by learners and their relation to the emotions that each learner expressed the most during the term.

In the visualization on the left of figure 4.7, each circle or globe corresponds to the emotion of a learner, and a set of concentric globes represents a learner. One different color is used for each different type of emotion. The occupied area is proportional to the level of the emotion. In some cases the circle is close to be of only one color, which means that this was the predominant emotion for a user . The position of the globe set on the X axis is defined by the final grade that the learner obtained in the course. Learners pass the course with a grade of 5.0, thus every learner on the left half of the square failed to pass while the right half passed the class successfully.

The Y axis is set according to the amount of events received from the learning environment. The closest a learner is to the bottom of the square, the less active she

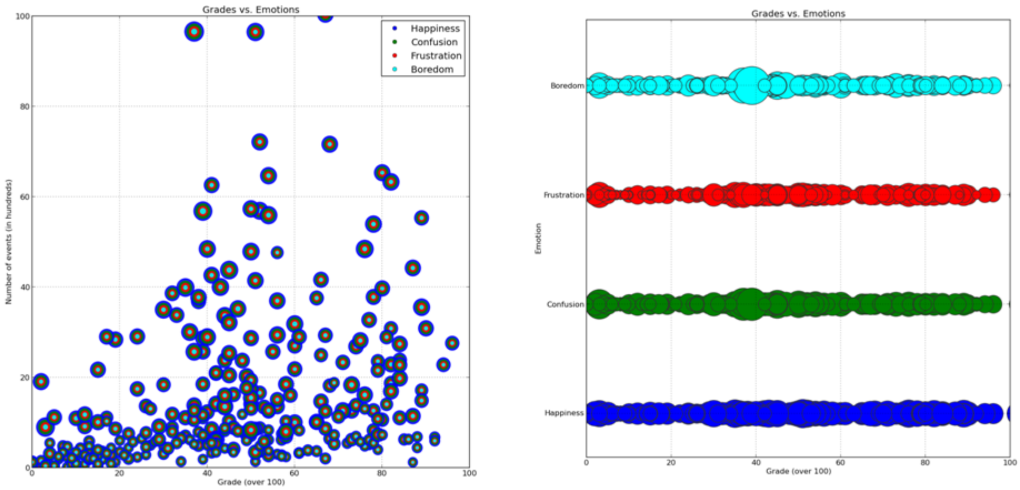

Figure 4.7: Contextual visualization adding the value of final mark.

was during the learning activities of the class. Learners at the top of the square are outliers that generated large amounts of events.

The separation between the ratios of the circles depends on the average level for each emotion. The set of circles is sorted by size in descendant order, meaning that those circles with a large cyan circle in the center have felt more confused in average than those with a small cyan circle. One example that illustrates this difference is the two outliers at the top. The confusion area on the one that failed to pass the course is rather larger than the one on the right, who actually passed the course.

A second version of the globes visualization is included on the right of Figure 4.7. The same information is contained in this representation but the number of events is not taken into account. The X axis is still associated to the final score of the learner in the course, and the size of the globes is also ruled by the average level for the given emotion and the given learner. There are two significant changes in this version. First, a learner is not represented by concentric circles anymore but by a set of circles aligned vertically. The second difference is that he position at Y axis is fixed according the reflected emotion.

From both of these last visualizations, it is interesting and also expected to see learners with a high level of confusion to fail the course. In general, it is also interesting

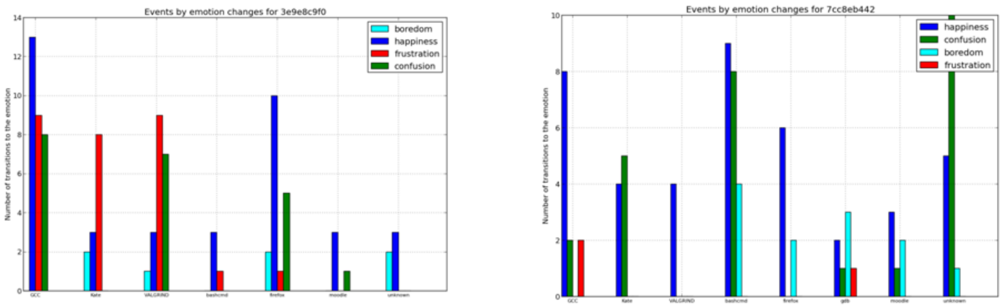

Figure 4.8: Visualization showing changes of affective state and the application causing it.

that for most learners the feeling of happiness was the highest one, disregarding whether they passed the class at the end of not. In addition, happiness and boredom are the predominant emotions for students with high scores and few events.

4.3 Visualizations of change in emotion

The third type of visualizations is related to changes of emotions during the learning activity. The visualization is this category (see Figure 4.8) is about the changes of emotions produced by each tool that the learner used. The example shows this information for the same learners of the figure 4.6. It is interesting to see that actually the memory profiler (Valgrind) generated most of the changes into the state of confusion, while the compiler (GCC) provoked changes to the state of happiness.

The learner on the right shows a different pattern although with some similarities. Compiler and web browser keep (firefox) on causing the learner fall into the happiness emotion, but here the web browser does not provoke to fall into the confusion state. In addition and unlike the first learner, command prompt (bashcmd) and text editor (Kate) are relevant tools for the learner to feel confused.

An additional characteristic of interest for the instructor is the constancy for each emotion expressed by the learner. The definition of constancy used in this work is the standard deviation of each emotion detected. An example of this visualization is provided in figure 4.9, composed of four panels, one for each of the histograms of the constancy level. The green histogram corresponds to the constancy level of confusion,

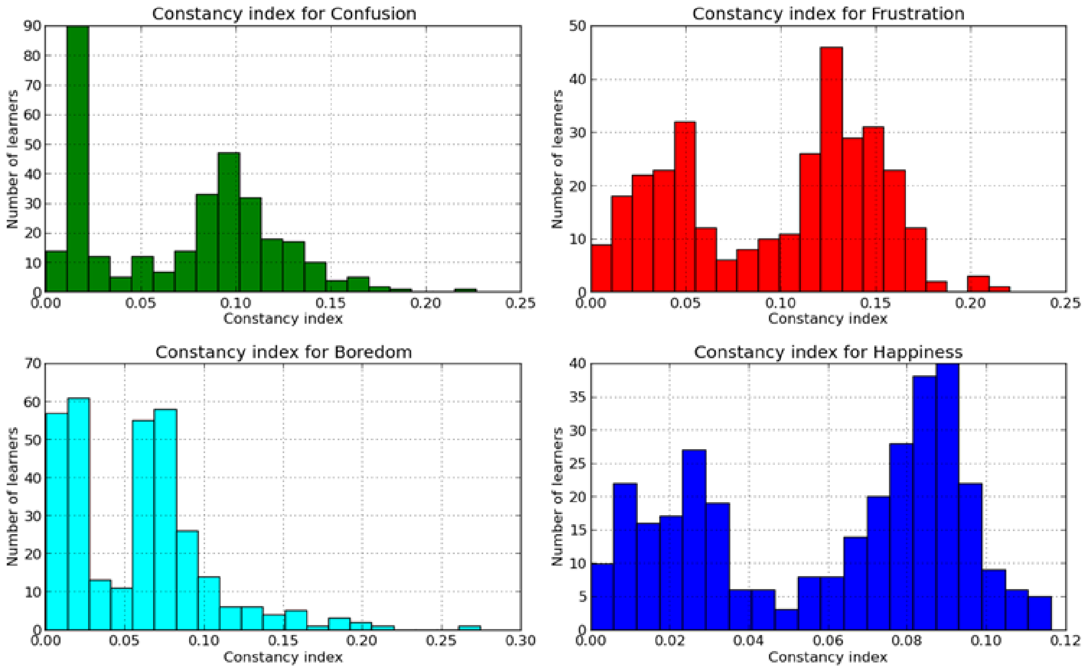

Figure 4.9: Visualization of constancy of each affective state.

and in this case it shows how many learners are indeed consistent on their level of confusion. A similar insight can be obtained from the boredom category, colored cyan. However, frustration and happiness (red and blue, respectively) indicate that the learner have fluctuated more than in the other emotions.

4.4 Visualizations of accumulated information

In this set we include four visualizations that represent the emotions of the whole class during the term. The objective of this kind of visualizations is to provide the instructor with a complete overview of the emotions expressed by learners in her class. This information is used to act on a specific learner but to analyze aspects that affect the group (e.g. learning material, environment and tools).

The first visualization is a pie graph of the average level of each emotion for the whole group of learners. The example (figure 4.10) shows that the average level is almost the same among the emotions. Happiness level is slightly higher than the rest of emotions, being the one frustration rather smaller.

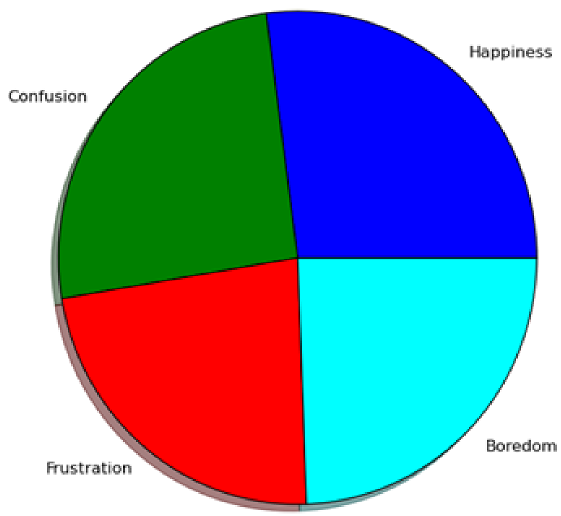

Figure 4.10: Pie chart showing average level of each emotion in the group of learners.

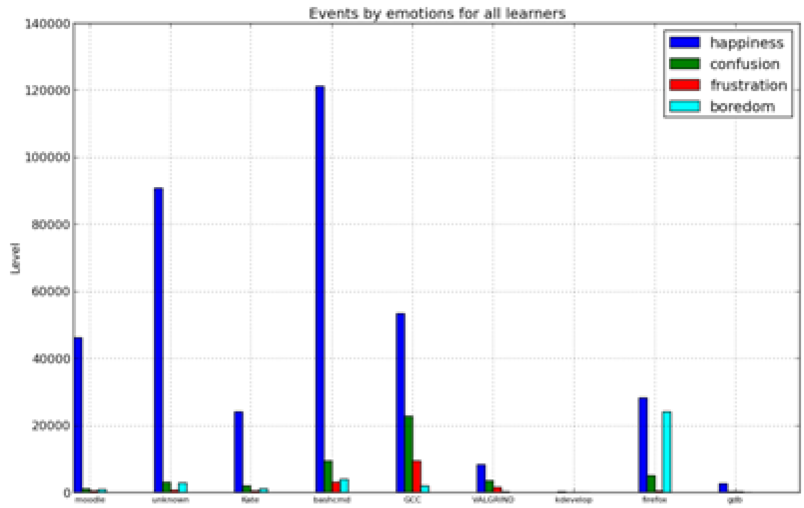

Figure 4.11: Aggregated amount of events for the group's affective states, by application.

The second visualization in this set shows the total amount of events from each tool and associated to each emotion. It is interesting to see that happiness is the predominant emotion during the observation of events. The command prompt is the tool mostly used during the course and also the one that generated the most events associated with happiness. As it could be foreseen, the tool that learners used the most while feeling bored was the web browser, giving that they used it to access external web content.

The next visualization is the average level of each emotion while using each tool in the learning activity. While the highest level of happiness is shown by the command prompt as in the previous visualization, there are other points worth to analyze. First, the level of happiness is maintained at a medium point by most of the tools, except memory profiler, development environment (kdevelop) and debugger (gdb). Although this could mean these tools had a negative impact on the learners' happiness in general, it should also be considered that the tools are also the least used (refer to figure 4.11). Compare for example the happiness average between the text editor (Kate, third from left to right) and the development environment (seventh from left to right); although learners do similar tasks on them, the text editor shows a considerable higher average level of happiness. Thus, the comparison is affected by the how much learners used each tool.

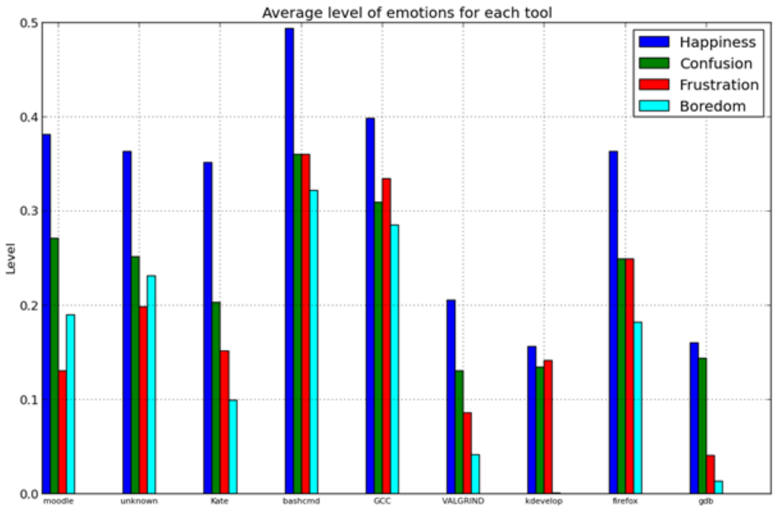

Figure 4.12: Aggregated average level of affective state by application.

Another interesting fact shown in figure 4.12 is the high level of confusion shared by tools like the LMS (moodle, first on the left) and the web browser. This indicates that learners tried to solve their programming errors by looking for explanations in the class material, and also by looking for information in any other web site.

As expected, command prompt and compiler are the tools that show the highest averages of confusion and frustration in learners. This fact relies also on the model used to infer emotions since having a problem while compiling a program is one of the events most associated to those emotions.

The last visualization (figure 4.13) is built using the same information than the previous one (i.e. the average level of each emotion when using each tool, for all learners). The difference relies on the information being presented in a radial format rather than linear. The learning tools are allocated around the circumference having four angular bars each. The ratio of each bar within the circumference is set by the average level of the associated emotion.

An improved version of the radial visualization is shown in Fig. 4.14. In a similar way, it uses a set of polar bars to present the average frequency of each learner affective state for each of the learning activities. Affective states of each learner are indicated through the color of each bar while labels are used to indicate the associated activity.

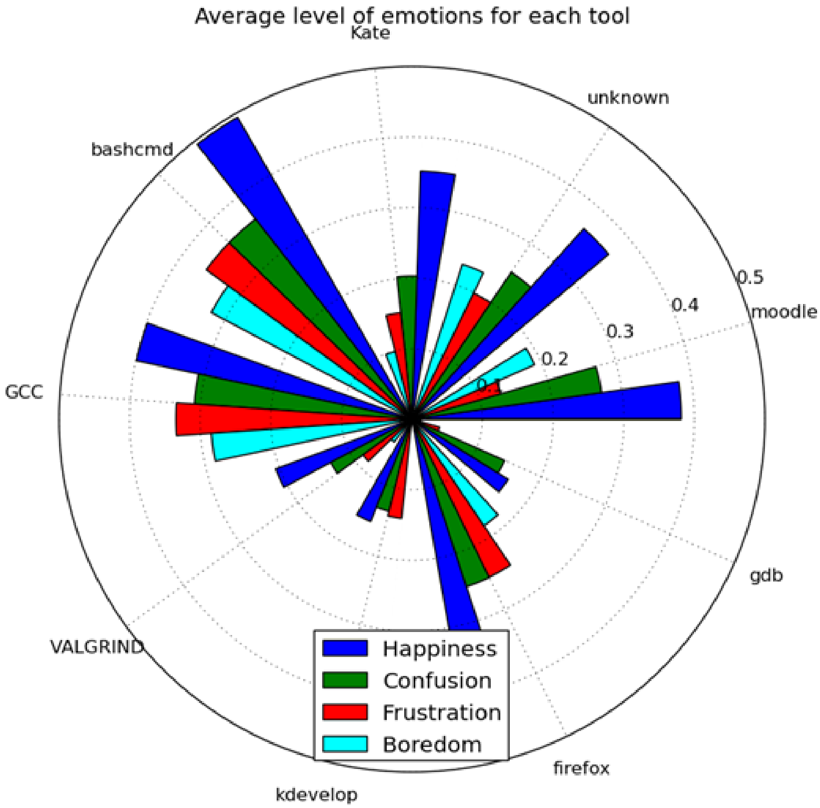

Figure 4.13: Average level of affective state by application.

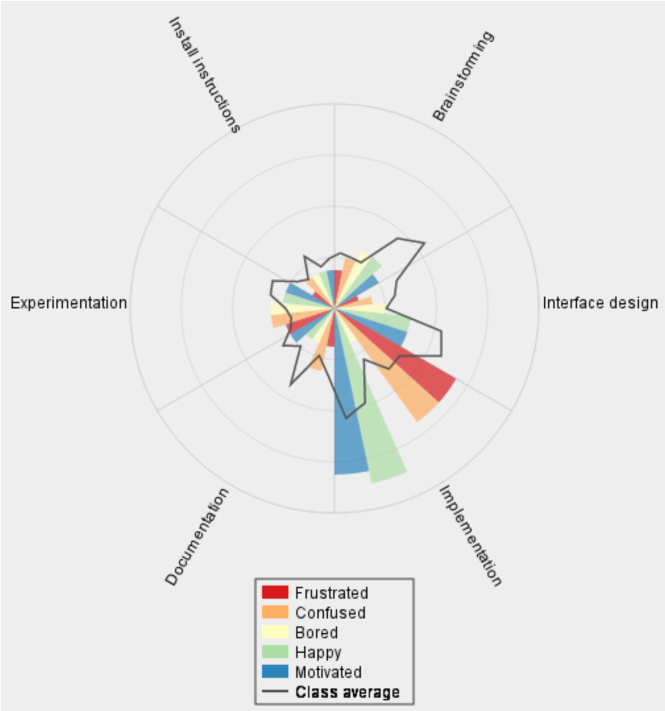

Figure 4.14: Visualization showing the frequency of each affective state for each activity. Polar bars show the values for the active student while solid line shows the values for the class average.

The solid line shows the average value of the class for each emotion and activity.

4.5 Discussion and limitations

We have presented four types of visualizations to provide awareness of the affective state of learners and we have applied them in the context of education in a programming course. These visualizations are high level information that is derived as a result of a processing from low level data of users' interactions with different technological educational tools. The visualizations are grouped in four categories: time-based, context-based, emotional changes, and accumulated information.

Time-based visualizations allow the instructor to analyze the changes of each emotion during the term of the class. The instructor is then able to see any pattern in the emotional changes of the learner and to know what caused the learner to change an emotion abruptly. An interesting use of these visualizations is to analyze academic and social activities that occurred when the changes of emotions appear. For example, emotion of learners is expected to change during the exam period of the university. Examples of social activities that could affect the emotion that learners show in class are relevant sport events or political announcements.

Context-based visualizations were presented as a way to analyze the effect of contextual elements onto a learner's emotions. Our proposals focused on two contextual elements: learning tools and final grades. Other options to be considered could be learning material in order to detect specific content that affects negatively the learning experience. Another important part of context is learner location localization, since this could provide valuable information about on how emotions are affected by doing a learning activity at home instead of the university. These new elements can also be analyzed in the context of changes of emotions and allow solving questions like: Does the place where the learning activity is done provoke a change of emotion? What place generates more frustration when a learner uses the compiler? Do learners get more bored at home than at the university?

Chapter 5

Evaluation

5.1 Evaluation of detection of emotions in a programming environment

5.1.1 Data collection

In order to collect data about the actions performed by students during a programming session, we used an updated version of the virtual machine presented by Romero-Zaldivar et al. [93]. The virtual machine has installed all of the tools that students will use in the laboratory activities of the programming subject. In addition, the virtual machine is configured so that it records the interaction of the students with a set of tools. The events generated by these interactions are stored into a set of log files within the virtual machine. Some of the recorded tools are specific for programming, such as: command line, compiler, debugger, and memory profiler; while others are of a broader use: web browser and text editor. The teaching staff provides this virtual machine to the students at the beginning of the term. Students are informed about the details of how events are captured and how to disable the process if wanted, in order to comply with local regulations for the treatment of personal information.

Students are instructed to submit their work to the teaching staff through a control repository system where they upload the programs and files created during the laboratory sessions and class projects. Along with their files, the virtual machine sends the log files to a centralized repository accessible to the teaching staff. Log files are then erased from the virtual machine, thus when students submit again only new events will be sent to the repository.

5.1.2 Mechanism to self-report affective states

In order to evaluate the emotions detected, it is needed to have a source of information about the real emotion that students were experiencing during the learning activity. We have followed an approach of having students report on their own emotions as done in prior studies [104, 43, 41]. We developed an application and installed it in the virtual machine for asking the student to indicate the level of a set of emotions. The interface of the application is a form with a set of sliders with negative and positive, thus each slider represents a pair of opposite emotions; this interface is similar to the proposed in [105]. The form asked about four pairs of emotions: happiness-sadness, interest-boredom, clearness- confusion and hope-frustration. The form appears recurrently during the work in the virtual machine and the student must change the value of the four sliders in order to close the form and continue working. The student responses are recorded in an additional file that is sent to the central repository as well. A screen capture of the form is provided in Fig. 5.1, where the level of happiness is 20%, the level of interest is 10%, clearness presents an 80% and hope a 60%.

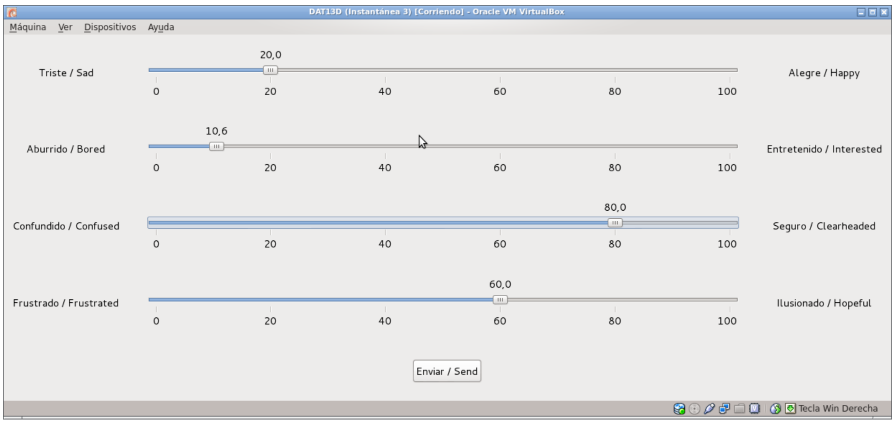

Figure 5.1: Capture of form to self-report affective states.

5.1.3 Experiment setting and participants

The experiment was conducted during an optative laboratory session of a programming course in the second year of the engineering program in a Spanish university. During the session, students performed a rehearsal of the laboratory final examination, meaning that students would follow the same instructions and use the same environment they would later use for the real examination. Thus, assessing the emotions of students during the session would not affect the marks they obtained in the subject. The session was repeated in four different time slots in order to reach the majority of the students.

The virtual machine was configured so that it would ask the student to indicate her emotions every 7 minutes. The rehearsal exam was designed to last between 30 and 45 minutes, which indicated that learners should provide between 4 and 6 responses to the emotion form. At the beginning of the session, students were informed about the description of the experiment and the procedure to follow once the emotion form appeared. Being a rehearsal of an examination, students were not allowed to communicate among themselves, review their notes nor ask subject-related questions to the teaching staff.

In order to participate in the sessions, students had to respond to an online form indicating the time slot they wanted to attend. A total of 62 students participated in the sessions and submitted their work. We filtered out the data of participants that did not fulfill minimum requirements for our analysis: 1) the virtual machine should have recorded at least 5 events of the student interaction with the computer and 2) the student should have answered to the emotion form a minimum of 4 times. After this filtering, the dataset was reduced to 41 students, with a gender distribution of 30 males and 11 females. Fig. 5.2 shows a histogram of the amount of events collected for each learner. It can be appreciated that most of the learners generated a very low number of events during their work, which is understandable given the duration of the activity.

The task of analyzing series of events generated by students can be labeled as a machine learning problem for the classification of sequences.

5.1.4 Analysis of emotion detection using continuous variables

First, students with a few number of events are discarded. The threshold is established in 10 events during the class session. These students with less than 10 events are no taken into account, which gives a total of 35 as the total number of students analyzed.

A repeated measures MANOVA analysis has been applied for the four considered emotions as dependent variables, taking into account two within-subjects factors: the type of detector (students' self-report using the form or with the Markov model) and the different instants of time in which the emotions were measured (there were 4 different

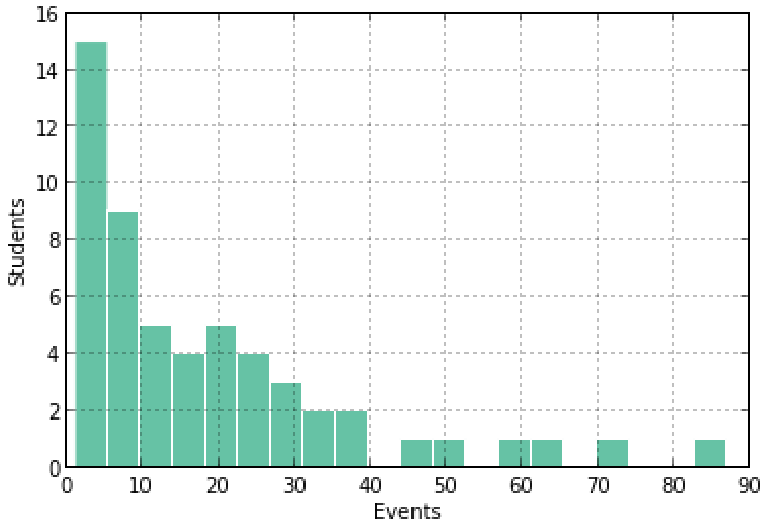

Figure 5.2: Histogram for the amount of events collected for each student

instants of times when students' emotions were retrieved). Results show that there is not a statistically significant difference on happiness, boredom, confusion and frustration over time and depending on the type of detector, $F(4, 31) = 1.35$, p=0.26, partial Eta Squared=0.41.

As the sphericity assumption is not meet, then the F factors should be corrected regarding the within-subjects effects. The Greenhouse–Geisser tests showed that there was not effect of the instant of time for happiness, $F() = 0.42$, p=0.63, partial Eta Squared=0.012, boredom, $F() = 1.307$, p=0.28, partial Eta Squared=0.037, confusion, $F() = 1.04$, p=0.35, partial Eta Squared=0.03, and frustration, $F() = 0.52$, p=0.56, partial Eta Squared=0.015. None of the pairwise comparisons between any instants of times was statistically significant for any of the considered emotions' variables. These results means that there were not significant differences on the means of emotions of the students along the class session, and the means of the emotions remained quite similar during the time.

Taking into account the effect of the type of detector of emotions, the Greenhouse–Geisser tests showed that there was a statistically significant difference on happiness, F=12.74, p=0.001, partial Eta Squared=0.273 and boredom, F=7.61, p=0.01, partial EtaSquared=0.273, but there was not effect on confusion, $F() = 0.15$, p=0.70, partial Eta Squared=0.004, and frustration, $F() = 0.02$, p=0.89, partial Eta Squared=0.001.

Therefore, the mean value of happiness and boredom was different depending on the detection using forms or with the computer based model based on Markov models. However, the mean value of confusion and frustration was not different depending on the type of detector. The pairwise comparisons between both types of detector for the four emotions give the following results of confidence intervals at 95Bonferroni correction are the following: [-21.56, -5.91] for happiness, [-28.24, -4.28] for boredom, [-7.88, 11.62] for confusion, [-11.08, 12.73] for frustration.

The confidence intervals denote that the difference of means of the emotions are not high, specially for the confusion and frustration. However, this fact does not guarantee that the detector based on the computer based Markov approach is close to the emotions reported by the students in forms. In order to check it, we apply correlational analysis in which we use the Pearson correlation for each instant of time and for each emotion between the values obtained by both types of detectors. Perfect alignment of detection of emotions would give a Pearson correlation of 1. However, the obtained correlational results are low, being less than 0.35 in all cases and even with negative values in some cases. Therefore, we can conclude that the detection of the self-reported values of emotions by students using the proposed model was not good for this specific environment and experience, and we were not able to detect the emotions reported by the students using an event based analysis of the students' interactions with the system.

5.1.5 Effect of affective states on final marks

Table 5.1 provides the regression index between affective states provided by the learners during each iteration of the questionaire and the final mark they obtained in the subject.

The correlation index (R) obtained indicates the null correspondance between the final mark of the learner and all emotions included in the study. Clearness and its counterpart, confusion, tend to be one of the emotions more correlated with changes of learning gains; however, the results of this analysis don't reflect this principle. It is interesting to observe that the higher correlation occurs on the hope emotion, although it is also counterintuitive to have a negative correlation index in this case.

5.1.6 Analysis of emotions detection with discrete values

An initial hypothesis to explain the low correlation between the inferred and real levels of each affective state was the fact that students had difficulties to assess their own

Table 5.1: Regression index and statistical significance between the final mark and students emotions

Emotion	Iteration	R	p
Happiness	1	0.077	0.593
Happiness	2	0.015	0.918
Happiness	3	-0.043	0.765
Happiness	4	-0.061	0.672
Interest	1	-0.046	0.753
Interest	2	-0.059	0.686
Interest	3	-0.056	0.698
Interest	4	-0.016	0.910
Clearness	1	0.068	0.638
Clearness	2	0.033	0.821
Clearness	3	0.044	0.764
Clearness	4	0.025	0.864
Hope	1	-0.179	0.214
Hope	2	-0.189	0.188
Hope	3	-0.143	0.323
Hope	4	-0.127	0.379

emotions in a continuous value. It was observed that students would not care about the specific numerical value they assigned to each state but to the range where the value landed. Thus, as a continuation of the evaluation, we discretisized the continuous values for both the inferred and real states by classifying each one in three fragments: up to 33% the negative state is present, up to 66% the neutral state is present and up to 100% the positive state is present.

Then the classification was evaluated through an analysis of Cohen's Kappa; the results of this analysis are presented in table 5.2. These results categorize the classification with a low performance.

Table 5.2: Cohen's Kappa analysis for each affective state at each of the four iterations

Iteration	Clearness	Happiness	Hope	Interest
1	0.091	0.074	0.04	0.019
2	0.194	0.071	-0.075	-0.027
3	0.194	-0.101	-0.246	-0.112
4	0.298	-0.082	-0.254	-0.143

The final hypothesis arising from this analysis was that the underlying data was not sufficient in order to infer correctly the affective state of students. Table 5.3 provides the results of the best algorithm for each emotion. The algorithms against which the results were compared are commonly used in the fields of Learning Analytics and Educational Data Mining. Examples of their application can be found in [32, 90, 42, 47]. In order to apply the algorithms on the collected data, it was needed to discretisize it in three stats: positive state, negative, and neutral. Then, the data was analyzed with each algorithm using the tool Weka[1].

Table 5.3: Results of applying algorithms commonly used in the literature

Emotion	Best Algorithm	Cohen's Kappa	ROC Area	F measure
Happiness	Random Forest	0.107	0.521	0.473
Interest	Naïve Bayes	0.080	0.542	0.490
Clearness	K*	0.095	0.51	0.398
Hope	Random Forest	0.0718	0.517	0.406

It can be observed that in all cases the Cohen's Kappa is classified as a low performing match which leads to the conclusion that the underlying data and variables cannot be used with the current tools to infer emotions in an effective way.

5.2 Evaluation of detection of emotions in the Khan Academy platform

In order to do a first evaluation of the models, we selected a set of learners that attended the initialization courses for the subjects of mathematics, physics and chemistry during the Summer of 2013. For each subject, we obtained the top 30 learners that had worked the most with platform exercises. Thus, some learners could repeat in the data set if their interaction with exercises was relatively high in more than one subject.

The four models described in the previous section were applied on the 90 students. Each learner's actions were fed into the implementation of each model, using time ranges of 10 minutes during the time periods of August 1st 2013 at 9:50 and August 8th 2013 at 11:10, which was a period with a high level of activity in the platform. In some cases, a learner would not interact as much as needed in order to infer his/her emotions, and was then removed from the resulting dataset. The final set consisted of 44 learners and

[1] http://www.cs.waikato.ac.nz/ml/weka/

the results of the models' application is analysed in the following subsections.

5.2.1 Evolution of emotions through time

The first analysis performed was the accumulated values for emotions for the whole data set of students. This analysis would give an overview of the evolution of emotions during the selected date range. The timelines for these values are shown in 5.3.

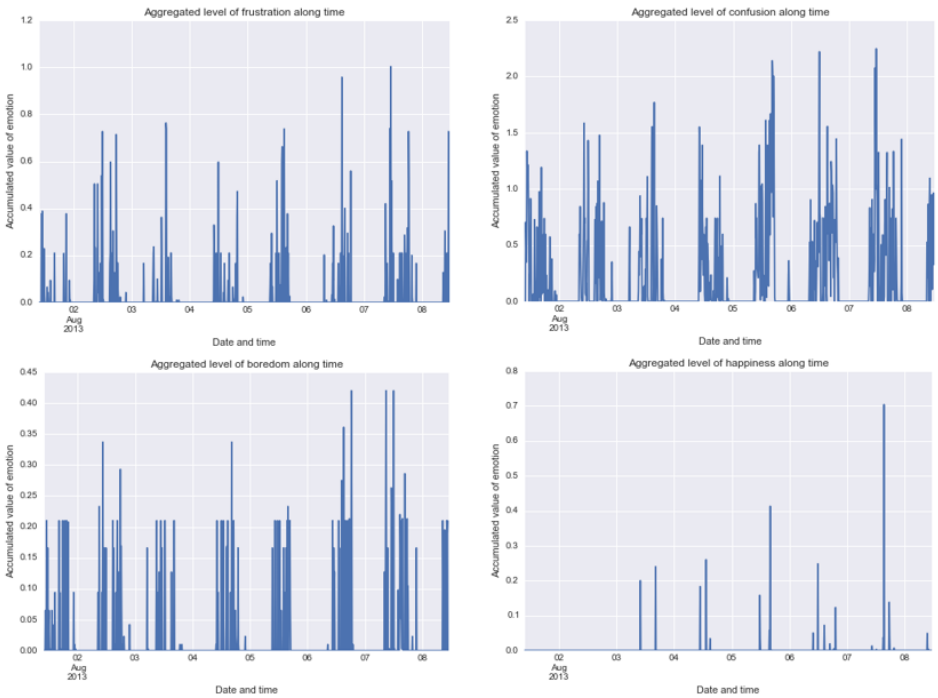

Figure 5.3: Timelines presenting the sum of emotion values for all students. Top to bottom and left to right: Frustration, Confusion, Boredom and Happiness

By comparing the top two graphics, it can be observed that values for frustration tend to be lower than those for confusion. This is an expected result and it coincides with previous work by D'Mello et al. where they observed transitions to frustration in extreme cases of confusion [36].

The accumulated boredom has also presented an expected evolution. While it is constant during most of the period, the first three days show a lower level while the last two days present higher values. The explanation for this behaviour is that students would be performing better at the end of the period and thus having skill levels higher than the needed for the exercises.

In a similar way, the accumulated happiness presents a slight increment along the analysed period. This is also foreseeable because learners are expected to be able to accumulate more badges as the course is advancing.

5.2.2 Accumulated emotions by hours of the day

In addition to the evolution of emotions along time, it is interesting to analyse patterns of learner's emotions fluctuations along the hours of the day. For this purpose, the values for emotions have been grouped and added by their hour of occurrence. The trend for each emotion is shown at 5.4 as a line chart.

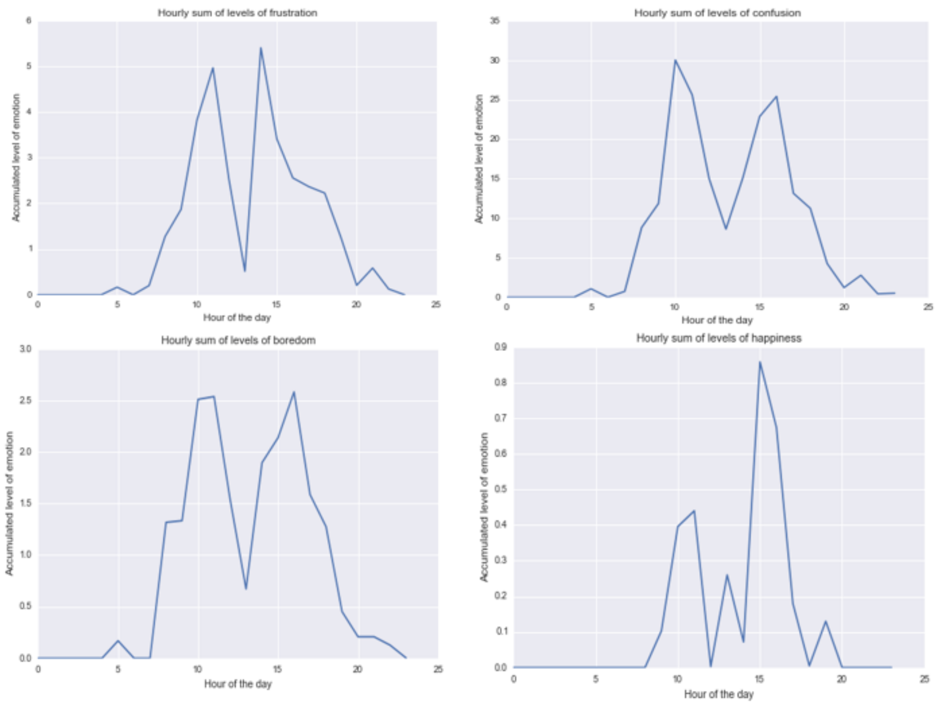

Figure 5.4: Accumulated values by hour of the day. From top to bottom and left to right: Frustration, Confusion, Boredom and Happiness

In this analysis, the models for frustration, confusion and boredom show a similar pattern marked by two peaks around 10:00 and 16:00. We can also observe that emotions have a null value during the night, from around 22:00 to 7:00, which is the expected behaviour. In a similar way, there is a local minimum around 13:00 for each emotion,

Table 5.4: Correlation indexes between the sums of emotion values for each learner and their evaluation metrics

Metric	Frustration	Confusion	Boredom	Happiness
Percentage of exercises solved well	0.490	0.440	0.335	0.151
Percentage of exercises with proficiency	0.607	0.581	0.361	0.241
Time spent in exercises	0.513	0.616	0.480	0.713

which would correspond to a lunch period. The trends in these three emotions also follow the sequence presented by D'Mello, in which confusion can be followed by frustration which then becomes boredom.

In the case of happiness, the values are higher during early afternoon. Our hypothesis in this case is that learners are continuing their morning sessions and thus obtaining badges as result of their day work.

5.2.3 Correlating emotions with course metrics

As a final analysis of the emotions inferred, it was of intended to evaluate the level of correlation between emotion levels and the evaluation metrics used in the courses. The selected metrics are the percentage of exercises that were solved correctly within each course, the percentage of exercises in which the learner reached a level of proficiency and the accumulated time spent on exercises during the course. These metrics are understood as independent from the emotion levels because, although most of the models take into account whether an exercise has been previously solved correctly, none of them use directly percentages of exercises done correctly.

The first step was to add up the calculated emotion levels for each student, thus having one value for each of the four emotions for each the 44 learners. The following process consisted on calculate the correlation index for each combination of an emotion and the selected evaluation metrics. Table 5.4 presents the correlation between the sums of emotions calculated every 10 minutes with the percentage of exercises in which learners reached proficiency.

A first observation of interest is the correlation index between frustration and the percentage of exercises with proficiency. Proficiency is not considered in any model, but there is a medium level of correlation with frustration (r = 0.607) which could indicate that learners were working hard in the exercises and even failing often before to obtaining proficiency.

Another interest factor appears in the correlation between the time spent in exercises and the emotions of confusion (r = 0.616) and happiness (r = 0.713). While the model for confusion does take into account the time that students are taking in order to solve exercises, the model for happiness omits this information and this concurs with the previous paragraph, indicating that learners spending larger amounts of time in exercises have also acquired more badges.

5.3 Evaluation of visualizations of emotions

5.3.1 User study 1

first user study, we evaluated two of the visualizations presented in chapter 4: the average emotion level per activity and the timeline. Furthermore, at this stage the timeline only presented the accumulated time dedication of the student and the class average.

The purpose of this user study was to perform an exploratory analysis of the developed visualizations. Thus, the feedback obtained from students would help to improve these visualizations.

Participants

This user study was conducted in the context of a course on information visualization at a graduate level (i.e., Master degree). First, students received theoretical and practical sessions about the different topics of the course. Next, and as part of the evaluation of the course, students had to do and present a project which included 12 types of activities: brainstorming, designing visualization, gathering data, parsing data collected, filtering and mining data, getting started with the visualization library D3.js, implementing the visualization, implementing interaction in the visualization, reading resources, reading research papers, preparing questions for paper and preparing research presentations. The project lasted five weeks, from late February to early April of 2014. The user study took place during this period.

As participants were registered for an information visualization course, they all had a relevant knowledge of principles and theories involved in the creation of visualizations. Thus, their feedback was highly interesting during the stage of early definition and development of the visualizations.

In total, 42 students were registered for the course. Out of these 42 students, 10 students participated in the first user study.

Data collection

We conducted 10 think-aloud sessions with one student at a time. The session was organized in three phases: 1) filling out a survey to capture data about their work during the project, 2) conducting tasks with the two visualizations, and 3) filling out an evaluation survey about the visualizations.

The survey to capture data about students' work during the project asked explicitly about the students' affective state for each type of activity done in the project. This way, for each type of activity, students had to indicate how frequently they have experienced the five affective dimensions known to occur in learning scenarios: motivation, happiness, boredom, confusion and frustration [106]. It also included a question about the amount of time dedicated to the project during the course for each of the weeks.

The evaluation survey included questions about the usability, the utility and the insights of the visualizations. The usability was measured with the System Usability Scale (SUS) method [107].

Students evaluated the utility through two 5-point Likert scale direct questions, rating the two visualizations from not useful at all to very useful.

Students were also asked about the utility of other information of interest that could be represented through visualizations. They could rate the utility of five types of information on a 5-point Likert scale: 1) types of used resources (e.g., forums, blogs or files), 2) detailed information of one student, 3) comparing actions between two students, 4) detailed statistics of most used resources and 5) information about content creation by students.

Insight was assessed through direct questions regarding information that visualizations intended to provide. The objective was to assess whether students can interpret the presented information. For instance, students were asked with a 5-point Likert scale whether their affective states were much below, below, on average, above or much above the average. They were also asked to indicate what their more frequent emotion was and the activity where they were more different compared to the other students. In addition, they had to select the time periods in which they had worked the most and the least.

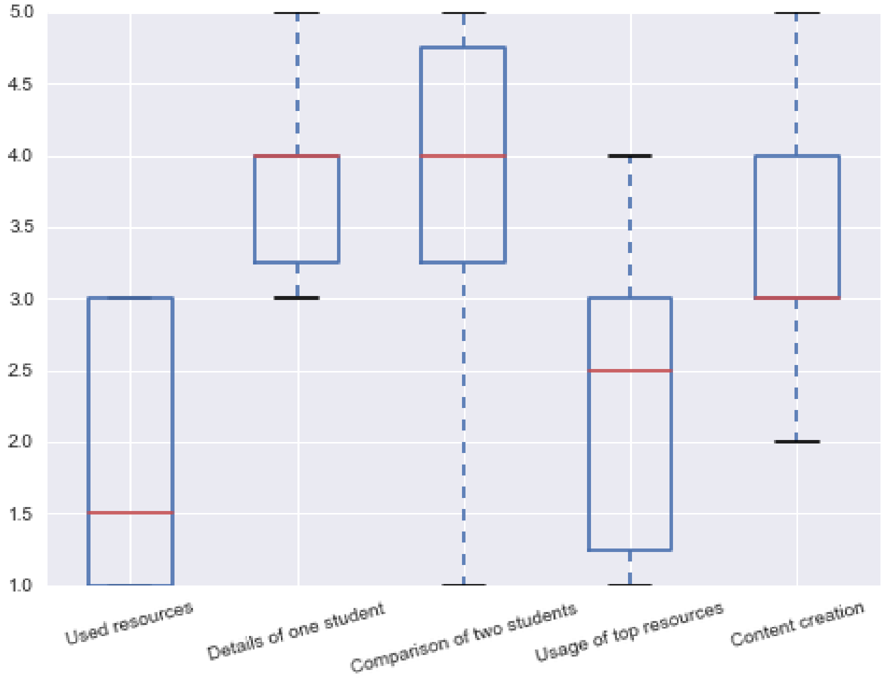

Figure 5.5: Frequency for answers given to the question "What other data would you like to have visualized or have accessible?".

Usability

The usability results obtained from the evaluation of the SUS questions averaged 72.5 points which can be assessed as good [108].

Analysis of usefulness

The timeline was the visualization perceived as most useful by the students, with three people grading it as very useful and seven as useful. The emotion per activity visualization was graded as useful by six students. The other students were less convinced by the usefulness (score three on a scale from one to five).

Regarding the information that was of interest for the students, detailed information of one student, comparing two students and information about content creation were the top priorities. The information related to types of used resources and the usage of top resources were the least prioritized. In Fig. 5.5 we present a set of box-plots illustrating the priorities given to each option.

Analysis of insight

The perceived differences of students' affective states with respect to the mean of the classroom based on the visualizations and the real differences in students' affective states and the mean was tested using the Pearson correlation with N=10, giving the following results: frustration (r=0.521, p=0.122), confusion (r=0.673, p=0.033), boredom (r=0.613, p=0.060), happiness (r=0.224, p=0.534), motivation (r=-0.623, p=0.054) and time dedication (r=0.737, p=0.015).

The correlation coefficients are high and positive in general which suggests that students' perceptions about the information of the visualizations were quite according to the reality. However, most of the results are not significant because the number of cases in the sample is too low (N=10). This might also explain the exception regarding the motivation, which gives a negative correlation or the happiness with a low correlation coefficient. User study 2 will analyze the same aspect with more cases which would enable the possibility to have significant results.

Other questions assessing the insight of the visualizations have been evaluated directly by comparing the value provided by the student with a correct answer obtained through data analysis. Table 5.5 lists these questions, the obtained results as a percentage of the correct answers and the number of categories that could be selected as an answer. The number of categories is important when analyzing the results of each question, as this is the number of possible options that the student can select. For example, there are 12 categories for question "The activity where I am most different from other students is" which implies a 8.33 percentage of guessing the right answer by chance. Twenty percent of the answers were correct, which is more than by chance. The analysis of the data from Table 5.5 shows that in general students interpreted well the visualizations having right answers greater than by chance. Students answered more times correctly when responding to questions related to the period where they worked the most and the least.

Interview results

The interviews provided valuable insights to improve both of the presented visualizations. The students commented about the difficulty to identify the value in some of the radial bars used in the visualization of affective states per activity. This was due to either having adjacent bars with similar colors or to a low level being undistinguishable in the

Table 5.5: Results of assessment of objective questions in final survey of user study 1.

Final Survey (10 responses)		
Question	Correct	Categories
My most frequent emotion during the class has been:	30%	5
The activity where I am most different from other students is:	20%	12
The period in which I worked the most has been:	90%	4
The period in which I worked the least has been:	70%	4

chart. Furthermore, some students weren't aware that the solid line represented the class average.

In general, students expressed that they "liked the timeline and the comparison with the class average". On the other hand, the visualization of affective states per activity generated was difficult to understand by some students with explanations like "it's hard to see the information of all students", "the red color (of bars representing frustration) is too distracting" and "it's confusing that bars don't start from zero".

Some students provided creative suggestions for further information to include in the visualizations. For instance, there was a suggestion to include the equivalent of "Return Over Investment" where the dedicated time would be the investment and the mark obtained the return. Another suggestion was to specify the type of task in which the time was dedicated (lectures, homework, studying and group work).

5.3.2 User study 2

For the second user study, we improved the two visualizations and added two new visualizations to address the comments of user study 1. The contrast of colors and the visibility of elements in both visualizations were improved. Interaction was added to clarify the details of the visualizations: the affective states per type of activity showed the value of each bar when the mouse cursor hovered over it. The timeline offered the option to hide and show data series, also including information about the affective states of the student.

For this user study, we implemented and evaluated two new visualizations with emphasis on individual and detailed information, as such information was identified as important by students of the first user study. These visualizations are the heat-map and scatter-plots described in chapter 4.

The purpose of this study was to evaluate the usability, usefulness and insight of the four visualizations. Two visualizations were also used in user study 1 but now with improvements done. These visualizations were available during the whole study while the last two were provided in the last week for a more elaborate user study.

Participants

The participants of this user study were 105 students of a first year of technological bachelors. The course subject was human-technology interaction and introduced all the concepts and processes involved in the design of usable interfaces for technological artefacts. At the end of the semester, students worked on a project about the design of a thermostat. The project duration was four weeks. At the end, students presented their project to the teaching staff and a subset of the classroom.

For this project, we defined six types of activities in which the students could be working: brainstorming, interface design, implementation, writing documentation, experiment with users and writing installation instructions.

Data collection

Once a week during the project, students were emailed with a link to an online survey and a link to a web application that showed the visualizations of their own emotion related data. The first weeks, just two visualizations were provided, whereas in the last week the four visualizations were provided. The reason for this is that some of the visualizations require data over a minimum period of time.

In order to maintain anonymity of information, students were asked to create a personal identifier. This identifier was included as a question in the survey and it was also required in order to access the visualizations. In order to avoid different students from creating the same identifier, we provided them with a specific format and restricted the use of trivial IDs such as "1234" or "ABCD".

Every week, students completed the following tasks: 1) filling out a survey about their activity during the week, 2) interact with several visualizations, and 3) filling out a survey about the visualizations.

The questions asked to the students about their activity were the following: students had to indicate how frequently they had experienced each affective state while performing each of the project activities and the time dedicated to the project. Students were allowed to report activity to a week different than the current one. After the data was submitted, the student could use a web application to access the visualizations.

Next, the data collected from these surveys was used in each visualization as follows:

- Visualization 1: Values for each affective emotion and each activity were shown directly. The average for all students was computed and shown on the solid line.
- Visualization 2: Values for the student were plotted according to the week they were provided. An average for the class was also included.
- Visualization 3: The intensity value of each cell represents the average value for the corresponding emotion at the given week.
- Visualization 4: For each affective state, the visualization plots the visualization value for each student along the date and time of the survey submissions.

In this study the students were also asked to answer an evaluation survey to assess the usability, usefulness and insight of the visualizations continuously. As in the final survey of the first user study, the usability was evaluated through SUS questions, while the usefulness was evaluated with additional 5-point Likert scales to rank each visualization from not useful at all to very useful. Finally, the questions used to assess objectively insight of the visualizations were the following:

- 5-point Likert scale to indicate whether the student is much below, below, average, above or much above the class for each emotion and time dedication.
- Indicate the most frequent emotion experienced during the project.
- Identify the activity that motivated students (the whole class) the most.
- Identify the activity that frustrated the student the most.
- Identify the activity where the student is most different from the rest.

In addition to weekly evaluations, a think-aloud session took place at the end of the course. The think-aloud session was conducted with batches of two to four groups, involving 6 to 12 participants in each session. The four visualizations were briefly explained and afterwards we asked students feedback and inquired about their interests. Section B in chapter A shows the questions asked in this final survey.

We received 298 submissions from 95 students for the data gathering survey, with 91% of the responses coming from male students and 9% from females. Most of the submissions (78.5%) belonged to students 20 years old or younger, 17.4% between 21 and 25, 1.7% between 26 and 30, 0.7% between 31 and 35, and 1.7% between 36 and 40. The survey for weekly evaluations received 218 from responses from 85 students, while 52 students participated in the final evaluation.

The methodology of data collection was different from user study 1. First, the SUS usability data is taken every week from the weekly survey and not at the end. Second, the visualizations presented to the students for the final survey in the think-aloud session took data from the weekly submitted data from students. Finally, usefulness and insight of the visualizations are evaluated with the survey of the last week plus the final survey, because it is only at the end when all the four visualizations are present. The survey questions are different from user study 1 (e.g. the questions of the final survey were not present in the user study 1) because there are two new visualizations which require new questions.

Usability

The average SUS score for the set of visualizations was 60.1. Fig. 5.6 presents a box-plot for each week of the study; as shown, the average usability stays constant over the weeks. The week was only provided in the data gathering survey, thus we inferred the week for the evaluation survey through a match of timestamps of both surveys. As indicated above, the heat-map and scatter-plots were introduced in the last week of the project and this can interfere with the evaluation of this week.

Usability results are less good than in the user case 1. The explanation of this should take into account the students' profile. In user study 1, students were experts in visualizations, whereas students in user study 2 have little or no knowledge about information visualization techniques.

Analysis of usefulness

Students also indicated the level of usefulness for each visualization using a 5-point Likert scale. The average usefulness score was 2.5, with the time line and the scatter-plots having a slight increment over the mean. Fig. 5.7 presents box-plot for the evaluation of each visualization.

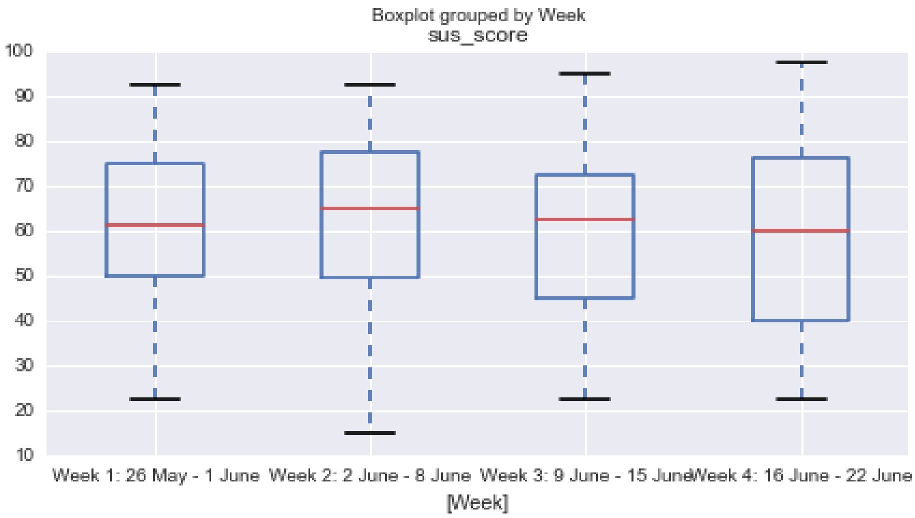

Figure 5.6: SUS scores obtained for all visualizations during the four weeks of the user study.

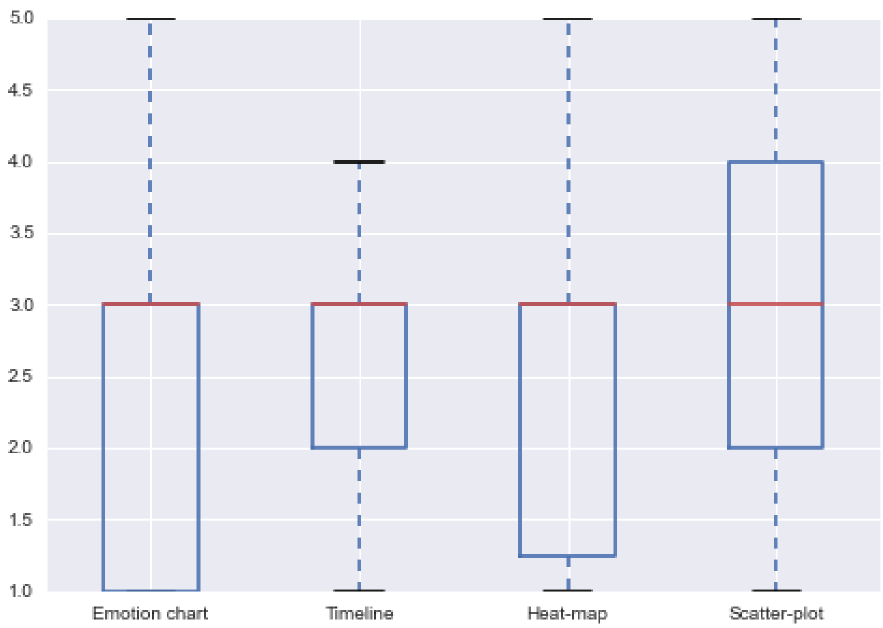

Figure 5.7: Usefulness marks from 1 to 5 for each visualization.

Analysis of insight

The perceived differences of students' affective states with respect to the mean of the classroom based on the visualizations and the real differences in students' affective states with respect to the classroom mean was tested using the Pearson correlation with N=34, giving the following results: frustration (r=0.634, p=0.000), confusion (r=0.620, p=0.000), boredom (r=0.551, p=0.000), happiness (r=0.684, p=0.0.000), motivation (r=0.829, p=0.000) and time dedication (r=0.374, p=0.040).

There is a statistically significance in all of them, having high levels of relationship in general with the exception of time dedication. This means that students were able to interpret correctly the provided visualizations in many occasions. However, there is room for improvement as the perfect understanding by all of the students would give correlation coefficients of 1.

Table 5.6 presents the results of the evaluation using objective questions.

Table 5.6: Percentage of correct responses to the objective questions in the survey.

Survey of last week (n=35)		
Question	Correct	Categories
Identify your most frequent emotion during the project.	48%	5
Identify the activity that motivated students the most.	23%	6
Identify activity that frustrated you the most.	69%	6
Identify activity where you differ the most from peers.	31%	6
Final survey (n=47)		
Question	Correct	Categories
How does your confusion evolve along time?	61%	3
Identify the week when students were more frustrated?	34%	4
Do you think your time dedication has an effect on your boredom?	53%	3

Results presented in Table 5.6 show that the percentage values of correct answers are considerable greater than by chance and give insights about the number of times that the learners interpreted correctly the visualizations for different aspects. The number of categories gives an idea of the number of possibilities a student can choose. The most options a student has, the most difficult for the student to get the right answer.

Therefore, the percentages of correct answers should be interpreted taking into account the number of different categories.

Some of the questions posed in these surveys are actually difficult to answer. For instance, the highest affective states (motivation and happiness) have a difference of 0.21 and the mean of each student's standard deviation of all states is 0.27. In some cases, a student would select an incorrect emotion as her most frequent but in fact the value of such emotion was very close to the highest one. The second question of the final survey presents a similar case: the correct answer obtained from the data analysis was week 4 but the amount of dedication in weeks 2 and 3 was similar. If we had considered all of these 3 options as valid, 98% of the answers would have been correct.

Interview results

Interview comments were very heterogeneous in the interviews of the second user study. For instance, the sense of usefulness varied among students. Some of them valued the affective state per activity:

> I liked the states per activity the most. After that will go the timeline, followed by the heat map. Finally the scatter plots.

Other students considered the timeline the most useful:

> The timeline is easiest to interpret, since it is in a form I am used to and since it doesn't contain that much data at the same time, which the others do. Especially the heat map and scatter plot are containing too much detailed and deviating information, which makes it hard to get an overview. The emotion per activity is okay, but also not readable very easily, because some coloured areas are very small and it is not always clear which colour is represented at what place of the grey line.

Others valued the combination of data and design used in complex visualizations like the heat map:

> It is difficult finding some meaningful values in the scatter plots. However the information in the heat map is grouped together nicely. Timeline shows a nice overview of how affective states progressed as well.

The teaching staff also provided valuable feedback about potential improvements. The main suggestion was to allow the instructor to indicate an expected amount of time dedication. This would allow learners to know whether they were dedicating less time than what the instructor was planning. The inclusion of this value would also allow the teaching staff to analyze whether the work load is being set appropriately for the current group of learners.

5.3.3 Discussion

First, there are some limitations to these user studies that should be articulated. Whereas we were able to assess usability, usefulness and insight with a relatively large number of students in user study 2, the number of students in user study 1 is too low to draw strong conclusions. Second, data collection was performed in a manual way in both user studies. Although there are studies on methods and techniques to capture and analyze emotion data of students in an automatic way [101, 109], the focus of this paper is on evaluating representation of such data with different visualization techniques. Whereas manual data acquisition works to evaluate the visualizations based on real student data, the acquisition process may have an influence on perceived usefulness and interpretation of data.

In general, usability results indicate that the visualizations are easy to use for students with knowledge of visualization techniques. A SUS score of 72.5 points can generally be assessed as good [108]. The same results could not be confirmed by user study 2. In this user study, students participated with little or no knowledge of information visualization techniques. The average SUS score in this user study is 60.1 - which indicates that students had difficulties with using the visualizations. This result indicates that visualization techniques need to be designed with care: some of the more complex visualizations included in the dashboard (heatmap, radial chart) may be a barrier for uptake by a general audience with no background in information visualization.

Usefulness results indicate that students perceive a simple timeline that represents time dedication and evolution of affective states over time as the most useful visualization. This visualization was rated as more useful that the affective states per activity visualization in user study 1. Also in the second case study with two other visualizations added, this visualization received the most positive results. Thus, a simple timeline visualization may be the most useful to support awareness of student data.

Insight was measured by correlations in the actual data and student perceptions. In addition, we measured how well students were able to interpret the visualizations with objective questions. There is a statistically significance for all the correlations, having high levels of correlations but less than 0.7 in general. This means that, in many occasions, students were able to interpret correctly the provided visualizations regarding the comparison of their affective states with respect to the mean. However, there is room for improvement as perfect awareness would give correlation values of 1. In addition, not all objective questions were answered well by students. Whereas straightforward questions were answered well, more difficult questions have relatively low correct responses (23% and 31% for some questions). Although still higher than by chance, further research is required to improve insight that can be derived from the visualizations - particularly for students with little or no knowledge of information visualization techniques.

In summary, results indicate that visualizations can be deployed in a useful way to support awareness and reflection of student data, but visualizations need to be designed with care to address the needs of students. Simple timeline visualization may have a higher value than more complex heatmap or radial visualizations. Which data to include in such visualization constitutes a further line of research. Whereas students of user study 1 indicated that they were interested in more detailed data about individual students, the representation of such data remains a challenge. Evaluation results of user study 2 indicate that the visualizations that we selected (heatmap, scatterplot with three dimensions) are too difficult to understand by users with no background in information visualization. Our future work will focus on simplifying these representations to enable use by a general audience.

Chapter 6

Recommendation based on affective information

6.1 Description of the generic architecture

The purpose of the proposed architecture is to deliver a set of recommended learning meta-data following a Software as a Service approach and based on affective information of the learner. The meta-data provides information about any element involved in a learning scenario: resources, activities, learners, instructors or feedback; we refer to all of these as *learning elements*. The architecture is composed by two layers: a service layer that executes the storage and recommendation tasks, and a client layer embedded in a learning environment. The details of each layer and their communication are described as follows.

6.1.1 Service layer

The service layer is in charge of receiving petitions from several clients and doing the requested tasks. The available tasks are to update the affective information of a learner, to update the information of a learning element, and to recommend learning element for a given learner. The first two tasks represent an administration interface for the management of learners' affective states and learning elements. Recommending learning elements is the main task of the service and it is the one that consumes most computational resources.

The service layer includes two storage elements to keep the information of the learning elements and the learners' affective states. The storage of learning elements is divided in categorical data-sources for each of the possible types of elements to store: users (learners or instructors, learning resources with content, learning activities, and feedback. These data sources only include meta-data and not actual content. This decision relies on the fact that the recommendation service is not meant to act as a repository of learning elements but just as a referrer.

The format of the database can be any usual specification for each specific type of element. For example, learning resources can make use of Learning Object Meta-data (LOM) or SCORM. The description of learning activities can be done with specifications like IMS Learning Design or Simple Sequencing. Plus, relationships between learners and instructor can be indicated through Friend-of-a-Friend (FOAF). The specifications of the format to use depend on the use case and are left to be decided by the implementation of the architecture.

The storage for learners' affective state is updated by requests from the client layer. The service is ready to create a new learner profile which consists of the learner identifier, current affective state and the record of the interactions with the elements to recommend and the affective state presented when that interaction occurred. The format used to define the affective state is decided by the implementation of the architecture as well. The specification EmotionML should be strongly considered because, although still being a W3C Candidate Recommendation, it allows flexible and complete definitions of affective states.

Besides the storage elements, the service layer has a recommendation engine cluster. The engine is designed as a cluster because the task to generate recommendations is the most expensive in terms of computational resources, which makes it the critical process to scale. The cluster contains as many instances of a recommending engine node as needed to support the service demand at the moment. Each node is in charge of analyzing the recommended elements, the affective states of a learner and generate assign a relevance score to each element the learner has not interacted yet. The algorithm used to calculate the recommendations is also left for the implementation of a specific use case.

6.1.2 Client layer

In the presented architecture, the client layer is represented by the learning environment and the instructor environment, since both can make use of the recommendation service to provide adaptation. The inclusion of recommendations is done by an embedded element deployed within each environment. The embedded elements communicate with the service layer to send and request information related to the recommendation based on affective states.

The element embedded in the learning environment sends the learner identifier and her affective state to the service layer. The client layer also informs to the service when a learner accesses a learning element. Optionally, the client can also be in charge of detecting the learning state of the learner and updates the affective state database.

There can be two moments when the embedded element requests information from the service layer. First, when accessing the learning environment the first time the client requests the last known affective state and the last recommended elements. The second moment is right after the affective status of the learner is updated. This is because a modification of the affective status implies a new set of recommendations for the learner, so the embedded element requests the list of recommended elements. After fetched, the list is displayed showing the title and description of each learning element. Then, the learner is able to select a resource based on its description or on the relevance score given by the recommendation engine. Figure 6.1, shows a diagram of the proposed architecture. The layers are displayed from bottom to top.

Being a generic architecture, it must consider a situation where elements are recommended to the instructor. This would be a case where the instructor receives information about learners that need an intervention, or feedback to provide to a specific learner. In this situation, the service layer connects not to a module embedded in the learning environment but in an environment that supports the instructor process. The functionalities of this embedded element are the same than the ones embedded in the learner environment, with the only difference of the user being the instructor.

The communication between the service and client layers is performed through the implementation of two Application Programming Interfaces (APIS): the Generic Recommender API and the Affective API whose purposes and methods are explained in detail as follows.

6.1.3 Generic Recommender API

The Generic Recommender Application Programming Interface (GRAPI) allows to manage the elements to recommend, and the interactions of the learners with the recommen-

Figure 6.1: Generic architecture of a affective learning recommender system based on the cloud.

dations. Each of these management functionalities is explained as follows.

1. Management of element. This part of the GRAPI is in charge of the creation of new elements to recommend as well of updating their information. The methods considered for this part are:

   CreateElement Creates a new element given its type (resource, activity or feedback), identifier, title, description and URL for fetching content if any, along with any other information.

   UpdateElement Updates the information of an existing element.

   RemoveElement This method deletes a element from the system.

   GetRecommendedElement It gets all the elements in the system that are recommended for a given learner.

2. Management learner interactions. This part of the GRAPI is in charge of the tracking the interactions between learners and the elements that they are recommended. The methods defined in this part are as follows:

   ViewElementStart Records the event of a learner starting viewing a recommended element.

   ViewElementStop Indicates the action of a learner stopping the viewing of a recommended element.

   MarkElement This method is used to indicate that the learner has marked a recommended element as favorite or relevant..

   ViewElementUrl Indicates that a learner has accessed the URL of a recommended element.

6.1.4 Affective API

The Affective Application Programming Interface (AAPI) allows the management of emotions, the present state of emotions for each user, and the evolution of emotions for each user during all the different actions in his learning process.

The AAPI can be divided into 4 different parts depending on their function:

1. Management of emotions. It is in charge of defining the different considered emotions. An emotion should be some aspect of the affective state of a person. Every emotion should have a value in the interval [0, 1]. For example, some types of emotions can be frustration or happiness. A value of 0 in happiness would mean that the user is very sad, while a value of 1 indicates that this user cannot be happier. The methods of the AAPI for this category are:

   CreateNewEmotion Creates a new type of emotion given by its name, description, and other possible information related to that emotion.

   UpdateExistingEmotion Updates the information of an existing emotion.

   RemoveExistingEmotion This method removes an existing emotion in the framework.

   GetExistingEmotions Gets all the defined emotions in the environment.

2. Management of learners. Defines the different learners with their information and related emotions to be considered. The methods of the AAPI for this category are:

   CreateNewLearner Creates a new learner with all her related information (name, surname, description, etc.). Other information such as the one related to IEEE-PAPI or IMS-LIP can be aggregated but would be part of another API but not of the affective one.

   UpdateExistingLearner Updates the information of an existing learner.

   RemoveExistingLearner Removes an existing learner.

   GetExistingLearners Get all the learners of the framework.

   AssignEmotionsToLearners With this method, it is possible to define or update the considered emotions that will be retrieved for a learner. For example, we could consider for some educative experience to retrieve happiness, frustration, and surprise, but for other context only consider one of them.

3. Updating of learners' emotions. This category enables to set the values of the emotions for each one of the learners during the time. For each learner, the affective storage will store learners current emotions and learners emotions when they accessed some learning resource or activity, received some feedback or interacted

with another user for requesting help or giving assistance. Therefore, there will be stored as many emotional states as the sum of contents, activities, requests for help and assistance for help that the learner interacted. The methods of the AAPI for this category are:

SetInitialEmotion This method will set all the values for all assigned emotions for a learner initially, i.e. before making any interaction in the learning system.

SetEmotionToLearnerInteraction This method will set all the values for all assigned emotions for a learner just after that learner finished the interaction with something or somebody in the learning process. The related information of the interaction should be stored and this should include the type of interaction (content, activity, request for help, assistance for others, feedback), the specific element which with they interacted, the time it took place, etc.

4. Retrieving of learners' emotions. This category allows to gather information about all the emotions of a learner while interacting in the learning environment. The methods of the A learner API for this category are:

GetEmotionalState This method allows to retrieve the values of the assigned emotions for a specific learner after his interaction with a specific content, activity, giving assistance, receiving feedback or requesting for help.

GetAllEmotions Provides all the assigned emotions of a learner during their learning process, giving the associated information of the learning content, activity, etc. related to each given emotion.

6.2 Use cases

In this section we describe three possible use cases that are supported by the generic architecture previously described. These are the recommendation of users, recommendation of activities, and the recommendation of learning resources which has been implemented as a proof of concept.

6.2.1 Recommendation of users

When a learner is blocked with some content or activity, e.g. he does not solve correctly an exercise or she does not understand well a specific theoretical content, this learner can request for help to other learners or even some teacher. In this case, it is a challenge to select the best person that can help and guide that learner. The issue is to recommend the most suitable user that can help the learner with problems.

Although typical information about the learning process is relevant for this recommendation such as the concepts and topics that a learner masters, however affective information about users is very relevant, as we might want to choose e.g. someone that is in a good mood in that moment so that she can offer the best help for the learner.

In our use case, when a learner requests for a recommendation of another user that can help himself in their learning process, the recommendation algorithm will take into account the following different aspects:

1. Measure of the convenience for being a user that gives assistance taking into account the present affective state. The idea is to find a user whose present affective state is good for helping another user. The different present values of the defined emotions will be taking into account with a weight according to their importance for this issue.

2. Measure of the level of knowledge of the user in the concepts of the resource. The idea is to find a user whose level of knowledge is adequate for the covered concepts in the element (content, activity, etc.) where the initial user was blocked. The knowledge level for each concept can also have a weight depending on its importance.

3. Measure of the similarity of the emotion for the same element. The idea is to select a user that has already passed for the same element where the learner is blocked and whose affective state in that moment was similar to the learner that is blocked. In that way, a learner that felt the same sensations can offer better help to this learner.

All of these 3 aspects will give a measure of adequateness and finally a global parameter will be given as a combination of these three aspects in a weighted way depending on their importance. In addition, if the final level of this parameter is not greater than a threshold, then a learner will not be recommended for giving assistance, but a teacher. This is because if it is considered that any learner did not have the minimum required level to provide help in that moment, then a teacher should act and intervene.

6.2.2 Recommendation of learning activities

The affective state of a learner is also a good indicator of the type of learning activity that the learner could perform. If the learner is bored, the recommender system could recommend activities that provoke an interaction such as solving a problem or discussing an issue with a group of classmates. However, if the learner is feeling confused, she could be recommended to read explanatory material related to the topics she is studying.

Learning activities can usually be associated to an affective state that the learner must present initially in order to carry it out. Thus, a rule-based algorithm is needed in order to recommend an activity that best fits the current situation of the learner. The aspects that take part of the condition to apply a rule will be the affective state, the learning profile (learning objectives, knowledge levels, etc.) and the history of activities already performed in order to satisfy possible dependencies. Once the learner completes the activity, her affective state and learning outcomes will be recorded. This will allow a validation of the recommendations and their effect on learning gains. In addition, this information will be valuable for future recommendations to similar learners.

6.2.3 Recommendation of learning resources

In this use case, the recommendation process within a recommender node follows the method known as user-based collaborative filtering. When a recommendation has to be done for a given learner, it first finds a set of the learner's *neighbors*, learners with similar patterns of access to resources. The level of similarity is to identify the neighbors of a learner is defined by a similarity function. In the case of the affective recommendation, the similarity of two learners is proportional to the amount of resources accessed by the learners when indicating the same affective state.

As the recommendations must take into account the affective state of the learner, the collaborative filtering process had to be extended to include that contextual information. The set of resources available for recommendation are a combination of the learning resources with the affective states. Thus,

$$R = L \times A$$

where R is the set of recommendable resources, L is the complete set of resources meta-data and A is the complete set of affective states. The recommendable resources are a tuple of a resource and an affective state.

After the learner's neighborhood is defined, learning resources are sorted based on how relevant they have been within that neighborhood of users. The resulting list must be filtered because it contains recommended tuples *(resource, affective state)* and the affective state might be any stored one. Thus, the list of recommended resources are only those that appear in a tuple where the affective state is the same one the learner presents at the moment.

The environment where we have implemented the affective recommender service is Amazon Elastic Computing Cloud (EC2), which is part of Amazon Web Services (AWS). EC2 allows us to define an image that acts as a blueprint to generate several instances of a computer with the same software configuration. Each one of these instances is what in the definition of the architecture we have called a node.

In our implementation, the storage element for learner resources meta-data is implemented as a database deployed in the engine MySQL. The same database implements the learner affective state storage element. Since the data might be accessed from many nodes of the cluster, the database engine is installed in an independent node, not meant to be part of the recommendation cluster. In order to manage the information stored in the database, the database node implements a RESTful web services, while the technology supporting the web application is J2EE.

The recommendation engine is developed on top of Apache Mahout machine learning engine. Mahout provides a set of libraries to implement machine learning models such as collaborative filtering, recommender systems, clustering, pattern mining and classifiers. Mahout is implemented in Java and this allows a straightforward integration with a web application developed with J2EE. As explained in [110], Mahout is conceived to be scalable through the framework for distributed processing Apache Hadoop, which allows the definition of clusters of computers with computational and storage capabilities.

The algorithms for collaborative filtering use the map/reduce paradigm. This paradigm consists in two processes that can be parallelized and distributed among several computers to increase their speed. The *map* process generates a sequence of pairs where usually the first element is an entity identifier and the second element is an entity characteristic that will be needed in a further computation. The *reduce* process receives the pairs generated by the map process and computes an incremental value associated with the entity. For example, the first step to identify the neighbors of a learner is to identify the most frequent occurrences of a pair *(affective state, learner resource)* for each user. In this case, the map process returns a pair with the syntax *(user identifier, (affective state, learner resource))*. Thus, the entity to identify is the user and the other item to include is the pair or affective state an resource. The reduce process receives the same pair and create an array for each received user. The elements of the array are complex structures with the syntax *((affective state, learner resource), count)*, so that by sorting the array in descendant order by the second element we obtain the top occurrences of pairs for the given learner.

The advantage of using the map/reduce paradigm is that both process can be done in parallel and in several computers. This allows the implementation to scale by just creating a new node with the same characteristics of a previous recommender node. Hadoop keeps control of the nodes that are available in the cluster to perform computational and storage tasks. Thus, we are provided with a simple way to auto-adjusting the size of the recommender cluster by just adding or removing nodes according to the service demand.

For the implementation of the client layer element, a widget has been developed as a proof-of-concept of a tool that interacts with the affective recommender service. The widget has been developed using the Software Development Kit (SDK) provided by ROLE Project [111]. ROLE aims to provide the learner with a framework to build her Personalized Learning Environment. The widget is implemented in JavaScript and HTML, and it follows the OpenSocial Gadget specification.

The widget interface has two functional sections represented by the tabs. Resources, the main tab, allows the learner to state her affective state from a static list provided by the recommendation service. Currently, the list is based on the affective states used by D'Mello et al. in [106]; these include frustrated, confused, bored, enthusiastic, motivated and the normal state, meaning that there is no relevant affective state at the moment.

Resources tab also presents a list of learning resources ordered by relevance for the learner in her current state. Thus, once the learner submits a change on her affective state, the widget send a recommendation request to the affective recommender service. When the response is received, the client analyzes the list of resources and embed their information as the list of recommended resources.

Second section is the *Profile* tab, where a time-line of the affective states reported by the learner is embedded. Its objective is to provide the learner with a visualization of her emotional changes during the learning activity being performed. The log of affective states is also provided by the learning resource service.

Finally, the Settings tab allows the learner to set her learning objectives. These might be changed during the learning activity, which also triggers a change of the learning resources that are recommended. Figure 6.2 presents a screen capture of the widget deployed in ROLE environment, with emphasis on the resources recommended to a frustrated learner.

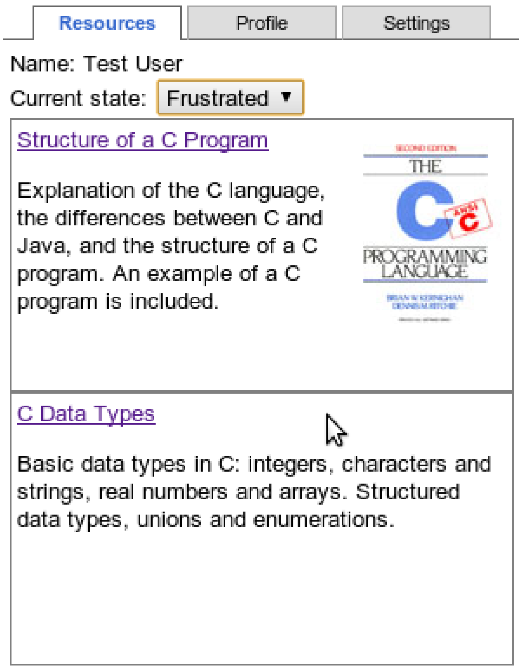

Figure 6.2: Implementation of the resource visualizer in the ROLE PLE.

A use case that exemplifies the use of the widget is the following. Alice, a university learner whose major is Computer Science, is trying to complete a C programming task that she was assigned as homework. She starts working highly motivated on the initial details of the program and she reflects be selecting the *Motivated* option among the affective states available in the widget. The widget communicates the affective state to the recommender service and this returns a list of resources suitable for Alice, such as C programming references and the user's manual of the compilation tool. Alice finds the resources helpful and uses them to write her code more quickly. As Alice advances she realizes that the task is not as easy as she first thought and that she might even encounter some programming errors that she was not expecting, this causes her affective state to change. When she find that she cannot fix a compilation error she starts feeling frustrated. Thus, she updates her affective state to *Frustrated*. Again, the widget communicates the new affective state to the recommender service, obtains the list of suitable resources and displays them as part of the widget content. The new list of recommended resources includes basic programming concepts and common programming errors with their respective solutions (see Figure 6.2). Alice accesses the resources and solves the error of her program. After seeing that the widget helps her along the task, Alice recovers her positive mood and continues her work to finish the homework.

6.3 Discussion and limitations

An advantage of the proposed architecture is the flexibility provided in two levels of the implementation. The first level corresponds to the possibility to recommend to the learner many kinds of elements like learner resources, activities or peers. The second level of flexibility is related to the algorithms used to identify the relevant elements. The generality of the architecture is exemplified with varied concrete use cases.

Throughout this section we have pointed the advantages of a cloud-based affective recommender system of learning elements. We have emphasized that the main benefit of using cloud technologies to deploy an affective recommender system is the gain of scalability.

Furthermore, there are some issues that must be taken into account as possible disadvantages of the architecture such as approaches to caching and privacy issues. The first issue is the difficulty to cache results in the presented architecture. Given that in a cloud recommendation service the computational workload is distributed among the nodes of the cluster, one node cannot cache the recommendation calculate in another node. This issue can be addressed either at the client layer, where the client itself would be in charge of caching the information of the recommended elements. The cache techniques to use in order to select the most important recommendations in this context is a relevant topic.

Another possible approach to solve this issue is through the inclusion of a middle layer that caches and dispatches the recommendations; this layer would be between the recommendation cluster and the interface of the service. Another issue are the formats to use (XML, RDF, etc.) in the framework for storing and exchanging the information. The proposed architecture is generic and admits different formats of the information. The selection of one format or another might depend on the formats that admits the tools that implement the different services of the API, or the defined level of interoperability with other external entities.

Another concern is related to privacy issues, since the affective state of the learner is stored in a database accessible from several recommender nodes. The key in this issue is that the recommending nodes are the only elements with permission to access the affective state information. Furthermore, the recommender service does not require to store information that identifies the learner immediately, such as the full name or email. Instead, the service can use a hashed identifier of the learner and that would not interfere with the process of recommending learning elements.

Chapter 7

Conclusions and future work

This chapter comments about the achievement of objectives, the publications related with this PhD, the research projects where this dissertation made a contribution and finally possible future work.

7.1 Achievement of objectives

This subsection will analyze the five initial objectives and their level of achievement, emphasizing the results related to each objective:

The first objective has been fulfilled with the definition of two different models for the detection of emotions with the only use of the students' events: one for a programming environment in which the students use different programming tools in order to learn C programming concepts. Another for a MOOC learning environment in the Khan Academy platform with intensive use of videos and exercises. For the first environment a Hidden Markov Model is proposed while a set of rules based on the actions is proposed for the second environment. Different pedagogical and sentiment ideas are mapped to both models for the detection of emotions. The emotions that were detected in both cases were: happiness, boredom, frustration and engagement.

The model of detection in the learning programming environment might be extended to be applied to other learning environments in which the students have to use and learn some tools. However, the way of mapping of events in the HMM should be modified taking into account the specific semantics of the learning environment. On the other hand, the model of detection in the MOOC environment might be extended to other

MOOCs where the semantics of exercises would be similar.

The accuracy of one of the detection models of emotions has been evaluated, this is the HMM model for the programming environment. The detection of emotions of the model in different instants of times was compared with respect to the students' self-reporting of emotions in a form in the same moments of time. The results showed that the detection of emotions was not good compared with the students' self-reporting. In addition, the application of other methods that were successful for ITS environments, did not result good compared with the students' self-reporting. The causes of this inaccuracy of the detection might be many, e.g. the detectors should be improved and the models that worked in other environments might not work in the programming environment, or students might not take seriously their own self-report of emotions or they might not be aware of their own emotions. In addition, the experiment took place in a lab for about 90 minutes. If we had more time for the experience, more events would be collected and the results might change. Moreover, a different evaluation method might be used. For example, instead of the self-report of emotions, other works used the expert reports about the emotions of the students, so the experts evaluated the students' emotion states in different instants of time.

The detection model of emotions for the Khan Academy platform was not able to be evaluated because students interacted with the platform at home but it is not an on-site activity, and it was considered very intrusive to ask them for their own emotions in a form in several moments of time.

The third objective was achieved completely as this work proposes a set of visualizations related to emotions and affective states. Some of these visualizations just involve emotions and affective states while others involve the relationship of these emotions and affective states with other common variables in learning environments such as the grades. Many of the proposed visualizations related with emotions and affective states are not tight to a specific learning environment but can be applied to a high variety of learning environments. Therefore, they are not only valid for the learning environments that we used for the detection of emotions for other objectives.

We have presented different types of visualizations to provide awareness of the affective state of learners and we have applied them in the context of education in a programming course. These visualizations are high level information that is derived as a result of a processing from low level data of users' interactions with different technological educational tools. The visualizations are grouped infour categories: time-based, context-based, emotional changes, and accumulated information. Time-based visualizations allow the instructor to analyze the changes of each emotion during the term of the class. The instructor is then able to see any pattern in the emotional changes of the learner and to know what caused the learner to change an emotion abruptly. An interesting use of these visualizations is to analyze academic and social activities that occurred when the changes of emotions appear. For example, emotion of learners is expected to change during the exam period of the university. Examples of social activities that could affect the emotion that learners show in class are relevant sport events or political announcements. Context-based visualizations were presented as a way to analyze the effect of contextual elements onto a learner's emotions. Our proposals focused on two contextual elements: learning tools and final grades.

For the fourth objective, we had to select a set of visualizations. It was not feasible to evaluate all the visualizations since the participants cannot spend all the time we desire, but only a reduced amount of time. The usability, usefulness and effectiveness of the considered visualizations was good for a group of users which are familiar with user interfaces, but was also nice for a group of participants who were not familiar with user interfaces.

Finally, the proposal and implementation of a framework for the recommendation was successfully achieved with the definition of the elements of the architecture, their relationships, an affective API, etc. and the definition of different user cases and the implementation of one of them for the recommendation based on the affective information.

7.2 Publications

The following publications are related with this dissertation:

1. Derick Leony, Pedro J. Muñoz-Merino, Carlos Delgado Kloos, Katrien Verbert, "Evaluating AffectVis, a visual information system for affective states in projects", Submitted to Information and Management

2. Derick Leony, Pedro J. Muñoz-Merino, José A. Ruipérez-Valiente, Abelardo Pardo, David Arellano Martín-Caro, Carlos Delgado Kloos, "Detection and evaluation of emotions in Massive Open Online Courses", Journal of Universal Computer Science, vol. 21, no. 5, pp. 638-655 (2015)

3. José A. Ruipérez-Valiente, Pedro J. Muñoz-Merino, Derick Leony, Carlos Delgado Kloos, "ALAS-KA: A learning analytics extension for better understanding the learning process in the Khan Academy platform", Computers in Human Behavior, vol. 47, pp. 139-148, (2015)

4. Derick Leony, Pedro J. Muñoz-Merino, Abelardo Pardo, José A. Ruipérez-Valiente, David Arellano, Carlos Delgado Kloos (2014), "Rule-based detection of emotions in the Khan Academy platform". International Workshop on Massive Open Online Courses. Antigua Guatemala, Guatemala

5. Derick Leony, Pedro J. Muñoz-Merino, Abelardo Pardo, Carlos Delgado Kloos, "Modelo basado en HMM para la detección de emociones a partir de interacciones durante el aprendizaje de desarrollo de software", Jornadas de Ingeniería Telemática, 2013

6. Derick Leony, Hugo A. Parada, Pedro J. Muñoz-Merino, Abelardo Pardo, Carlos Delgado Kloos, "A generic architecture for an affective learning recommender system", Journal of Universal Computer Science, vol. 19, no. 14, pp. 2075-2092 (2013)

7. Carlos Delgado Kloos, Abelardo Pardo, Pedro J. Muñoz-Merino, Israel Gutierrez, Derick Leony, "Learning Analytics at UC3M", IEEE Educon Conference, 2013

8. Derick Leony, Pedro J. Muñoz-Merino, Abelardo Pardo, Carlos Delgado Kloos, "Provision of awareness of learners' emotions through visualizations in a computer interaction-based environment", Expert Systems With Applications, vo. 40, no. 13, pp. 5093-5100 (2013)

9. Derick Leony, Abelardo Pardo, Hugo Alexer Parada, Carlos Delgado Kloos, "A Cloud-based Architecture for an Affective Recommender System of Learning Resources", Proceedings of the 1st International Workshop on Cloud Education Environments (2012)

7.3 Research projects

The results of this dissertation has been applied partially to the following research projects:

1. EEE Project - Espacios Educativos Especulares. (TIN2011-28308-C03-03)
2. eMadrid - Investigación y desarrollo de tecnologías para el e-learning en la Comunidad de Madrid. (S2009/TIC-1650)
3. iCOPER - Interoperable Content for Performance in a Competency-driven Society. (ECP-2007-EDU-417007)
4. Flexo - Desarrollo de aprendizaje adaptativo y accesible en sistemas de código abierto. (TSI-020301-2008-19)

7.4 Future work

Related to the five objectives, different improvements and future works can be thought.

Regarding the models of detection, they can be modified to try to improve the detection of emotions and affective states. For example, the HMM model for the detection in the programming learning environment might be changed by another probabilistic model such as the Analytical Hierarchy Model (AHM) to test if better results can be achieved. Moreover, the rule-based detection model applied to the MOOC environment can be improved with the inclusion of new data. So far, data about the students' interactions with exercises is only used, but it can be extended to include e.g. data students' interactions with videos.

In addition, new learning environments (such as different MOOC platforms) might be explored and detectors of emotions for such environments might be defined.

The way of validation of the accuracy of the prediction of the affective states might be changed. In this work, we used students' self-reporting of their affective states. But this might not be accurate enough as students might not take the activity seriously or they might not be aware of their own emotions and affective states. Other methodologies such as emotion detection with sensors or reports from external experts about the emotions of students might be better and can be object of future work.

The detection model of emotions in the MOOC environment was not validated because it was not feasible to ask the students of the experience about their emotions or measuring their emotions in another way. As future work, new experiences might be designed to validate the accurateness of the detection model in a MOOC environment.

Other types of visualizations might be considered such as learning material in order to detect specific content that affects negatively the learning experience. Another important part of context is learner location localization, since this could provide valuable information about on how emotions are affected by doing a learning activity at home instead of the university. These new elements can also be analyzed in the context of changes of emotions and allow solving questions like: Does the place where the learning activity is done provoke a change of emotion? What place generates more frustration when a learner uses the compiler? Do learners get more bored at home than at the university? We are also interested on designing new visualizations by enhancing or by combining the ones proposed in this work. This approach could include also the exploration of including dynamic or interactive elements into the visualizations. Thus, the visualization could use movement to represent variations in emotional information along dimensions such as time, person, learning tool, and learner location.

Another possible line of work is the implementation of these visualizations into a system accessible to instructors through the whole term. In this line, the chosen approach could be to implement several modules into Gradient Learning Analytics System (LearnGLASS) [112].

New evaluations of the visualizations might be considered with different groups of people. In addition, the evaluation of the visualizations which were not evaluated before might be considered. For example, one evaluation might be performed by experts in the area of emotions in learning scenarios and another one with instructors of the course represented in the visualizations.

Finally, regarding the recommender system framework, future work consists on evaluating the performance of the implementation presented in this work. The evaluation objectives might be too broad such as analyzing the improvement on the response time of a recommendation request to the server. Part of this work includes to analyze the correlation between the service performance and the amount of recommender nodes in the cluster; this would lead to a set of guidelines for deploying the recommender service in a real learning scenario. In order to address the response time problem, a solution might be to process in advance in the background future recommendations. A challenge here is how to determine the states that are more probable in the future, so that the recommendations can be available in advance, avoiding the response time problem. This issue is similar to the one discussed in the architecture in [113].

Another line of work consists in the development of plug-ins to include sensors as a method to populate the affective state database. Specifically we are working with sensors for galvanic skin response and the recognition of face gestures through a video camera. These sensors have been proven to detect affective states with accuracy [Picard, 2000] and thus might be improve the recommendation process. They would also allow the learner to focus on the retrieval and use of elements rather than constantly informing her current affective state. On the same track, it is also intended to improve the interface for the learner to provide her affective state. Several approaches will be taken in order to obtain contextual information about what provoked a given emotion in the learner and how did the recommendation of elements interferes her affective state.

Chapter 8

Bibliography

[1] Doug Clow. The learning analytics cycle: closing the loop effectively. In *Proceedings of the 2nd international conference on learning analytics and knowledge*, pages 134–138. ACM, 2012.

[2] Cristobal Romero and Sebastian Ventura. Educational data mining: A survey from 1995 to 2005. *Expert systems with applications*, 33(1):135–146, 2007.

[3] Barry J Zimmerman. Self-regulated learning and academic achievement: An overview. *Educational psychologist*, 25(1):3–17, 1990.

[4] Peter Brusilovsky, Charalampos Karagiannidis, and Demetrios Sampson. Layered evaluation of adaptive learning systems. *International Journal of Continuing Engineering Education and Life Long Learning*, 14(4-5):402–421, 2004.

[5] Robert A. Sottilare, Arthur Graesser, Xiangen Hu, and Heather Holden. *Design Recommendations for Intelligent Tutoring Systems: Volume 1 - Learner Modeling*. Design Recommendations for Intelligent Tutoring Systems. U.S. Army Research Laboratory, 2013.

[6] Peter Brusilovsky and Christoph Peylo. Adaptive and intelligent web-based educational systems. *Int. J. Artif. Intell. Ed.*, 13(2-4):159–172, April 2003.

[7] Fathi Essalmi, Leila Jemni Ben Ayed, Mohamed Jemni, Kinshuk, and Sabine Graf. A fully personalization strategy of e-learning scenarios. *Computers in Human Behavior*, 26(4):581 – 591, 2010. Emerging and Scripted Roles in Computer-supported Collaborative Learning.

[8] Pedro J. Muñoz-Merino, Manuel Fernández Molina, Mario Muñoz-Organero, and Carlos Delgado Kloos. An adaptive and innovative question-driven competition-based intelligent tutoring system for learning. *Expert Systems with Applications*, 39(8):6932 – 6948, 2012.

[9] Pedro J. Muñoz-Merino, Carlos Delgado Kloos, Mario Muñoz Organero, and Abelardo Pardo. A software engineering model for the development of adaptation rules and its application in a hinting adaptive e-learning system. *Comput. Sci. Inf. Syst.*, 12(1):203–231, 2015.

[10] Mario Muñoz Organero, Carlos Delgado Kloos, and Pedro J. Muñoz Merino. Personalized service-oriented e-learning environments. *Internet Computing, IEEE*, 14(2):62–67, March 2010.

[11] Pythagoras Karampiperis and Demetrios Sampson. Adaptive learning resources sequencing in educational hypermedia systems. *Educational Technology & Society*, 8(4):128–147, 2005.

[12] Kenneth R. Koedinger, Sidney D'Mello, Elizabeth A. McLaughlin, Zachary A. Pardos, and Carolyn P. Rosé. Data mining and education. *Wiley Interdisciplinary Reviews: Cognitive Science*, 6(4):333–353, 2015.

[13] Michel C. Desmarais and Ryan S.J.d. Baker. A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction*, 22(1-2):9–38, 2012.

[14] Thomas A. Warm. A primer of Item Response Theory. Technical Report 940279. *ERIC*, 1978.

[15] Michal Barla, Mária Bieliková, Anna Bou Ezzeddinne, Tomáš Kramár, Marián Šimko, and Oto Vozár. On the impact of adaptive test question selection for learning efficiency. *Computers & Education*, 55(2):846 – 857, 2010.

[16] Jeff Johns, Sridhar Mahadevan, and Beverly Woolf. Estimating student proficiency using an item response theory model. In Mitsuru Ikeda, KevinD. Ashley, and Tak-Wai Chan, editors, *Intelligent Tutoring Systems*, volume 4053 of *Lecture Notes in Computer Science*, pages 473–480. Springer Berlin Heidelberg, 2006.

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

[18] 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.

[19] Zachary A. Pardos, Neil T. Heffernan, Brigham Anderson, and Cristina L. Heffernan. Using fine-grained skill models to fit student performance with bayesian networks. *Handbook of Educational Data Mining*, pages 417–426, 2010.

[20] Jean-Claude Falmagne, Eric Cosyn, Jean-Paul Doignon, and Nicolas Thiéry. The assessment of knowledge, in theory and in practice. In Rokia Missaoui and Jürg Schmidt, editors, *Formal Concept Analysis*, volume 3874 of *Lecture Notes in Computer Science*, pages 61–79. Springer Berlin Heidelberg, 2006.

[21] Juergen Heller, Christina Steiner, Cord Hockemeyer, and Dietrich Albert. Competence-based knowledge structures for personalised learning. *International Journal on E-Learning*, 5(1):75–88, 2006.

[22] Pedro J. Muñoz Merino, José A. Ruipérez Valiente, Carlos Alario-Hoyos, Mar Pérez-Sanagustín, and Carlos Delgado Kloos. Precise effectiveness strategy for analyzing the effectiveness of students with educational resources and activities in MOOCs. *Computers in Human Behavior*, 47:108–118, 2015.

[23] Nicola Henze, Peter Dolog, and Wolfgang Nejdl. Reasoning and ontologies for personalized e-learning in the semantic web. *Educational Technology & Society*, 7:82–97, 2004.

[24] Vincent Aleven, Bruce McLaren, Ido Roll, and Kenneth Koedinger. Toward tutoring help seeking. In James C. Lester, RosaMaria Vicari, and Fábio Paraguaçu, editors, *Intelligent Tutoring Systems*, volume 3220 of *Lecture Notes in Computer Science*, pages 227–239. Springer Berlin Heidelberg, 2004.

[25] Pedro J. Muñoz Merino, José A. Ruipérez Valiente, and Carlos Delgado Kloos. Inferring higher level learning information from low level data for the khan academy platform. In *Proceedings of the Third International Conference on Learning Analytics and Knowledge*, LAK '13, pages 112–116, New York, NY, USA, 2013. ACM.

[26] Curtis R. Henrie, Lisa R. Halverson, and Charles R. Graham. Measuring student engagement in technology-mediated learning: A review. *Computers & Education*, 90:36 – 53, 2015.

[27] Genaro Rebolledo-Mendez, Benedict du Boulay, and Rosemary Luckin. Motivating the learner: An empirical evaluation. In Mitsuru Ikeda, KevinD. Ashley, and Tak-Wai Chan, editors, *Intelligent Tutoring Systems*, volume 4053 of *Lecture Notes in Computer Science*, pages 545–554. Springer Berlin Heidelberg, 2006.

[28] Valeria De Palo, Maria Sinatra, Giancarlo Tanucci, Lucia Monacis, Pierpaolo Di Bitonto, Teresa Roselli, and Veronica Rossano. How cognitive styles affect the e-learning process. In *Advanced Learning Technologies (ICALT), 2012 IEEE 12th International Conference on*, pages 359–363, July 2012.

[29] Yavuz Akbulut and Cigdem Suzan Cardak. Adaptive educational hypermedia accommodating learning styles: A content analysis of publications from 2000 to 2011. *Computers & Education*, 58(2):835 – 842, 2012.

[30] Jia-Jiunn Lo, Ya-Chen Chan, and Shiou-Wen Yeh. Designing an adaptive web-based learning system based on students' cognitive styles identified online. *Computers & Education*, 58(1):209 – 222, 2012.

[31] S. Alkhuraiji, B. Cheetham, and O. Bamasak. Dynamic adaptive mechanism in learning management system based on learning styles. In *Advanced Learning Technologies (ICALT), 2011 11th IEEE International Conference on*, pages 215–217, July 2011.

[32] Ryan S. J. d. Baker, Sidney K. D'Mello, Ma.Mercedes T. Rodrigo, and Arthur C. Graesser. Better to be frustrated than bored: The incidence, persistence, and impact of learners' cognitive-affective states during interactions with three different computer-based learning environments. *Int. J. Hum.-Comput. Stud.*, 68(4):223–241, April 2010.

[33] I-Han Hsiao, Fedor Bakalov, Peter Brusilovsky, and Birgitta König-Ries. Open social student modeling: Visualizing student models with parallel introspectiveviews. In *Proceedings of the 19th International Conference on User Modeling, Adaption, and Personalization*, UMAP'11, pages 171–182, Berlin, Heidelberg, 2011. Springer-Verlag.

[34] Rosalind W. Picard. *Affective computing*. The MIT Press, 2000.

[35] Paul Ekman. An argument for basic emotions. *Cognition & emotion*, 6(3-4):169–200, 1992.

[36] Sidney D'Mello. Monitoring affective trajectories during complex learning. In NorbertM. Seel, editor, *Encyclopedia of the Sciences of Learning*, pages 2325–2328. Springer US, 2012.

[37] Kate Forbes-riley. Predicting emotion in spoken dialogue from multiple knowledge sources. In *In Proceeedings of HLTNAACL 2004*, pages 201–208, 2004.

[38] Cristina Conati, Romain Chabbal, and Heather Maclaren. A study on using biometric sensors for monitoring user emotions in educational games, 2003.

[39] Liping Shen, Minjuan Wang, and Ruimin Shen. Affective e-Learning: Using Emotional Data to Improve Learning in Pervasive Learning Environment. *Educational Technology and Society*, 12(2), 2009.

[40] Varun Bajaj and RamBilas Pachori. Detection of human emotions using features based on the multiwavelet transform of eeg signals. In Aboul Ella Hassanien and Ahmad Taher Azar, editors, *Brain-Computer Interfaces*, volume 74 of *Intelligent Systems Reference Library*, pages 215–240. Springer International Publishing, 2015.

[41] Jennifer Sabourin, Bradford Mott, and JamesC. Lester. Modeling learner affect with theoretically grounded dynamic bayesian networks. In Sidney D'Mello, Arthur Graesser, Björn Schuller, and Jean-Claude Martin, editors, *Affective Computing and Intelligent Interaction*, volume 6974 of *Lecture Notes in Computer Science*, pages 286–295. Springer Berlin Heidelberg, 2011.

[42] Sidney K. D'Mello, Scotty D. Craig, Jeremiah Sullins, and Arthur C. Graesser. Predicting affective states expressed through an emote-aloud procedure from AutoTutor's mixed-initiative dialogue. *International Journal of Artificial Intelligence in Education*, 16(1):3–28, 2006.

[43] Ivon Arroyo, David G. Cooper, Winslow Burleson, Beverly Park Woolf, Kasia Muldner, and Robert Christopherson. Emotion sensors go to school. In *Proceedings*

of the 2009 Conference on Artificial Intelligence in Education: Building Learning Systems That Care: From Knowledge Representation to Affective Modelling, pages 17–24, Amsterdam, The Netherlands, The Netherlands, 2009. IOS Press.

[44] Ryan S. Baker, Sujith M. Gowda, Michael Wixon, Jessica Kalka, Angela Z. Wagner, Aatish Salvi, Vincent Aleven, Gail W. Kusbit, Jaclyn Ocumpaugh, and Lisa Rossi. Towards Sensor-Free Affect Detection in Cognitive Tutor Algebra. In *International Conference on Educational Data Mining*, pages 126–133. -, June 2012.

[45] Scotty D. Craig, Arthur C. Graesser, Jeremiah Sullins, and Barry Gholson. Affect and learning: an exploratory look into the role of affect in learning with autotutor. *Journal of Educational Media*, 29:241–250, 2004.

[46] Ryan Shaun Baker, Albert T. Corbett, Kenneth R. Koedinger, and Angela Z. Wagner. Off-task behavior in the cognitive tutor classroom: When students "game the system". In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 383–390, New York, NY, USA, 2004. ACM.

[47] Zachary A. Pardos, Ryan S. J. D. Baker, Maria O. C. Z. San Pedro, Sujith M. Gowda, and Supreeth M. Gowda. Affective states and state tests: Investigating how affect throughout the school year predicts end of year learning outcomes. In *Proceedings of the Third International Conference on Learning Analytics and Knowledge*, LAK '13, pages 117–124, New York, NY, USA, 2013. ACM.

[48] Diane Marie C. Lee, Ma. Mercedes T. Rodrigo, Ryan S. J. D. Baker, Jessica O. Sugay, and Andrei Coronel. Exploring the relationship between novice programmer confusion and achievement. In *Proceedings of the 4th International Conference on Affective Computing and Intelligent Interaction - Volume Part I*, ACII'11, pages 175–184, Berlin, Heidelberg, 2011. Springer-Verlag.

[49] Paulo Blikstein. Using learning analytics to assess students' behavior in open-ended programming tasks. In *Proceedings of the 1st International Conference on Learning Analytics and Knowledge*, LAK '11, pages 110–116, New York, NY, USA, 2011. ACM.

[50] Riccardo Mazza and Christian Milani. Exploring usage analysis in learning systems: Gaining insights from visualisations. In *In AIED Workshops (AIED'05)*, 2005.

[51] Hangjin Zhang, Kevin Almeroth, Allan Knight, Monica Bulger, and Richard Mayer. Moodog: Tracking students' online learning activities. In *World Conference on Educational Multimedia, Hypermedia and Telecommunications*, pages 4415–4422, 2007.

[52] Sten Govaerts, Katrien Verbert, Erik Duval, and Abelardo Pardo. The Student Activity Meter for Awareness and Self-reflection. In *ACM SIGCHI International Conference on Human Factors in Computing Systems*, 2012.

[53] Derick Leony, Abelardo Pardo, Hugo A. Parada G., and Carlos Delgado Kloos. A widget to recommend learning resources based on the learner affective state. In *Proceedings of the 2nd and 3rd International Workshops on Motivational and Affective Aspects*, 2012.

[54] Jose Luis Santos, Sten Govaerts, Katrien Verbert, and Erik Duval. Goal-oriented visualizations of activity tracking: A case study with engineering students. In *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*, LAK '12, pages 143–152. ACM, 2012.

[55] Hans-Christian Schmitz, Maren Scheffel, Martin Friedrich, Marco Jahn, Katja Niemann, and Martin Wolpers. Camera for ple. In *Learning in the Synergy of Multiple Disciplines*, pages 507–520. Springer, 2009.

[56] Riccardo Mazza and Vania Dimitrova. Coursevis: A graphical student monitoring tool for supporting instructors in web-based distance courses. *International Journal of Human-Computer Studies*, 65(2):125–139, 2007.

[57] Riccardo Mazza and Luca Botturi. Monitoring an online course with the gismo tool: A case study. *Journal of Interactive Learning Research*, 18(2):251–265, 2007.

[58] Ourania Petropoulou, Katerina Kasimatis, Ioannis Dimopoulos, and Symeon Retalis. Lae-r: A new learning analytics tool in moodle for assessing students' performance. *Bulletin of the IEEE Technical Committee on Learning Technology*, 16(1):1–13, 2014.

[59] Angel Juan, Thanasis Daradoumis, Javier Faulin, Fatos Xhafa, et al. Developing an information system for monitoring student's activity in online collaborative

learning. In *Complex, Intelligent and Software Intensive Systems, 2008. CISIS 2008. International Conference on*, pages 270–275. IEEE, 2008.

[60] Christopher Brooks, Jim Greer, and Carl Gutwin. The data-assisted approach to building intelligent technology-enhanced learning environments. In *Learning Analytics*, pages 123–156. Springer, 2014.

[61] Leah P Macfadyen and Shane Dawson. Mining lms data to develop an "early warning system" for educators: A proof of concept. *Computers & Education*, 54(2):588–599, 2010.

[62] Anna Lea Dyckhoff, Dennis Zielke, Mareike Bültmann, Mohamed Amine Chatti, and Ulrik Schroeder. Design and implementation of a learning analytics toolkit for teachers. *Journal of Educational Technology & Society*, 15(3):58–76, 2012.

[63] Elena Gaudioso, Felix Hernandez-Del-Olmo, and Miguel Montero. Enhancing e-learning through teacher support: two experiences. *Education, IEEE Transactions on*, 52(1):109–115, 2009.

[64] Javier Melero, Davinia Hernández-Leo, Jing Sun, Patricia Santos, and Josep Blat. How was the activity? a visualization support for a case of location-based learning design. *British Journal of Educational Technology*, 46(2):317–329, 2015.

[65] Erik Duval. Attention please!: learning analytics for visualization and recommendation. In *Proceedings of the 1st International Conference on Learning Analytics and Knowledge*, pages 9–17. ACM, 2011.

[66] Katrien Verbert, Sten Govaerts, Erik Duval, Jose Luis Santos, Frans Van Assche, Gonzalo Parra, and Joris Klerkx. Learning dashboards: an overview and future research opportunities. *Personal and Ubiquitous Computing*, 18(6):1499–1514, 2014.

[67] Kristin Stephens-Martinez, Marti A Hearst, and Armando Fox. Monitoring MOOCs: which information sources do instructors value? In *Proceedings of the first ACM conference on Learning@ scale conference*, pages 79–88. ACM, 2014.

[68] Pinelopi Krystalli, Panagiotis Arvanitis, and Panagiotis Panagiotidis. Evaluating serious games for foreign language learning: An online grading and visualization tool. *International Journal for Cross-Disciplinary Subjects in Education (IJCDSE)*, 5(1):1564–1570, March 2014.

[69] Jennifer L. Chiu and Marcia C. Linn. Supporting knowledge integration in chemistry with a visualization-enhanced inquiry unit. *Journal of Science Education and Technology*, 23(1):37–58, 2014.

[70] Siti Rosminah Md Derus and Ahmad Zamzuri Mohamad Ali. Integration of visualization techniques and active learning strategy in learning computer programming: a proposed framework. *International Journal on New Trends in Education and Their Implications*, 5(1):93–103, 2014.

[71] Shu-Ling Shieh and I-En Liao. A new approach for data clustering and visualization using self-organizing maps. *Expert Systems with Applications*, 39(15):11924–11933, 2012.

[72] Flora S. Tsai. Dimensionality reduction techniques for blog visualization. *Expert Systems with Applications*, 38(3):2766–2773, 2011.

[73] Feng Tian, Huisan Zhang, Longzhuang Li, Qinghua Zheng, and Yang Yang. Visualizing e-learner emotion, topic, and group structure in chinese interactive texts. In *Advanced Learning Technologies (ICALT), 2011 11th IEEE International Conference on*, pages 587–589. IEEE, 2011.

[74] Uroš Krčadinac, Jelena Jovanović, and Vladan Devedžić. Visualizing the affective structure of students interaction. In *Hybrid Learning*, pages 23–34. Springer, 2012.

[75] Nikos Manouselis, Hendrik Drachsler, Katrien Verbert, and Erik Duval. Recommender systems for learning. *SpringerBriefs in Computer Science*, 2012.

[76] Sushil Keswani, Allan Krans, Elizabeth Hedstrom Henlin, Jillian Mirandi, Matt Casey, and Ken Gagnon. 4Q11 Public Cloud Market Landscape. Coud business quarterly, Technology Business Research, Inc., April 2012.

[77] Declan Dagger, Alexander O'Connor, Seamus Lawless, Eddie Walsh, and Vincent P. Wade. Service-oriented elearning platforms: From monolithic systems to flexible services. *IEEE Internet Computing*, 11(3):28–35, 2007.

[78] Mario. Muñoz Organero, Pedro J. Munoz-Merino, and Carlos Delgado Kloos. Personalized service-oriented e-learning environments. *IEEE Internet Computing*, 14(2):i62–67, 2010.

[79] Luis M. Vaquero, Luis Rodero-merino, Juan Caceres, and Maik Lindner. A Break in the Clouds: Towards a Cloud Definition. *ACM SIGCOMM Computer Communication Review*, 39(1):50–55, 2009.

[80] Neal Leavitt. Is cloud computing really ready for prime time? *Computer*, 42(1):15–20, jan 2009.

[81] Mikroyannidis Alexander. A Semantic framework for cloud learning environments. *In: Chao, Lee ed. Cloud Computing for Teaching and Learning: Strategies and Implementation. Hershey, PA: IGI Global*, pages 17–31, April 2012.

[82] OKI (Open Knowledge Initiative), 2012. Last visited January 2013.

[83] Peter Brusilovsky. Knowledgetree: A distributed architecture for adaptivee-learning. In *Proceedings of the 13t international World Wide Web conference*, pages 104–111, 2004.

[84] Peter Dolog, Rita Gavriloaie, Wolfgang Nejdl, and Jan Brase. Integrating adaptive hypermedia techniques and open rdf-based environments. In *Proc. of The Twelfth International World Wide Web Conference, WWW 2003*, pages 88–98, 2003.

[85] Xiaodi Huang. An E-learning System Architecture based on Cloud Computing. *International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE)*, 2:74–78, 2012.

[86] Mohammed Al-zoube. E-Learning on the Cloud. *International Arab Journal of e-Technology*, 1(2):58–64, 2009.

[87] Sarah Stein, Jennifer Ware, Johanne Laboy, and Henry E. Schaffer. Improving K-12 pedagogy via a Cloud designed for education. *International Journal of Information Management*, http://dx.doi.org/10.1016/j.ijinfomgt.2012.07.009 2012.

[88] Arjun Arora Deepanshu Madan, Suneet Kumar. E-learning based on Cloud Computing. *International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE)*, 2(2), 2012.

[89] Katrien Verbert, Nikos Manouselis, Xavier Ochoa, Martin Wolpers, Hendrik Drachsler, Ivana Bosnic, and Erik Duval. Context-aware recommender systems

for learning: a survey and future challenges. *Learning Technologies, IEEE Transactions on*, 5(4):318–335, 2012.

[90] RyanS.J.d. Baker, Antonija Mitrović, and Moffat Mathews. Detecting gaming the system in constraint-based tutors. In Paul De Bra, Alfred Kobsa, and David Chin, editors, *User Modeling, Adaptation, and Personalization*, volume 6075 of *Lecture Notes in Computer Science*, pages 267–278. Springer Berlin Heidelberg, 2010.

[91] Abelardo Pardo and Carlos Delgado Kloos. Stepping out of the box. Towards analytics outside the Learning Management System. In *International Conference on Learning Analytics and Knowledge*, 2011.

[92] Martin Wolpers, Jad Najjar, Katrien Verbert, and Erik Duval. Tracking actual usage: the attention metadata approach. *Journal of Technology Education & Society*, 10(3):106–121, 2007.

[93] Vicente Arturo Romero Zaldívar, Abelardo Pardo, Daniel Burgos, and Carlos Delgado Kloos. Monitoring Student Progress Using Virtual Appliances : A Case Study. *Computers & Education*, 58(4):1058–1067, 2012.

[94] Ryan S.J.d. Baker, Sidney K. D'Mello, Ma.Mercedes T. Rodrigo, and Arthur C. Graesser. Better to be frustrated than bored: The incidence, persistence, and impact of learners' cognitive–affective states during interactions with three different computer-based learning environments. *International Journal of Human-Computer Studies*, 68(4):223 – 241, 2010.

[95] Khe Foon Hew and Wing Sum Cheung. Students' and instructors' use of massive open online courses (MOOCs): Motivations and challenges. *Educational Research Review*, 12:45–58, 2014.

[96] Nishikant Sonwalkar. The first adaptive MOOC: A case study on pedagogy framework and scalable cloud architecture—part i. In *MOOCs Forum*, volume 1, pages 22–29. Mary Ann Liebert, Inc. 140 Huguenot Street, 3rd Floor New Rochelle, NY 10801 USA, 2013.

[97] René F. Kizilcec, Chris Piech, and Emily Schneider. Deconstructing disengagement: analyzing learner subpopulations in massive open online courses. In *Pro-*

ceedings of the third international conference on learning analytics and knowledge, pages 170–179. ACM, 2013.

[98] Ilona Nawrot and Antoine Doucet. Building engagement for MOOC students: introducing support for time management on online learning platforms. In *Proceedings of the companion publication of the 23rd international conference on World wide web companion*, pages 1077–1082. International World Wide Web Conferences Steering Committee, 2014.

[99] José A. Ruipérez-Valiente, Pedro J. Muñoz-Merino, and Carlos Delgado Kloos. An architecture for extending the learning analytics support in the khan academy framework. In *Proceedings of the First International Conference on Technological Ecosystem for Enhancing Multiculturality*, pages 277–284. ACM, 2013.

[100] Rafael A. Calvo and Sidney D'Mello. Affect detection: An interdisciplinary review of models, methods, and their applications. *Affective Computing, IEEE Transactions on*, 1(1):18–37, 2010.

[101] Derick Leony, Pedro J. Muñoz-Merino, Abelardo Pardo, and Carlos Delgado Kloos. Modelo basado en hmm para la detección de emociones a partir de interacciones durante el aprendizaje de desarrollo de software. In *XI Jornadas de Ingeniería Telemática*, 2013.

[102] Ira J. Roseman, Martin S. Spindel, and Paul E. Jose. Appraisals of emotion-eliciting events: Testing a theory of discrete emotions. *Journal of Personality and Social Psychology*, 59(5):899, 1990.

[103] José A. Ruipérez-Valiente, Pedro J. Muñoz-Merino, Derick Leony, and Carlos Delgado Kloos. Alas-ka: A learning analytics extension for better understanding the learning process in the khan academy platform. *Computers in Human Behavior*, pages –, 2015.

[104] Cristina Conati and Xiaoming Zhou. Modeling students' emotions from cognitive appraisal in educational games. In *Intelligent tutoring systems*, pages 944–954. Springer, 2002.

[105] Amine Trabelsi and Claude Frasson. The emotional machine: A machine learning approach to online prediction of user's emotion and intensity. In *Advanced Learning*

Technologies (ICALT), 2010 IEEE 10th International Conference on, pages 613–617. IEEE, 2010.

[106] Sidney D'Mello, Rosalind Picard, and Arthur Graesser. Towards an affect-sensitive autotutor. *IEEE Intelligent Systems*, 22(4):53–61, 2007.

[107] John Brooke. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189:194, 1996.

[108] Aaron Bangor, Philip T Kortum, and James T Miller. An empirical evaluation of the system usability scale. *Intl. Journal of Human–Computer Interaction*, 24(6):574–594, 2008.

[109] Derick Leony, Pedro J. Muñoz-Merino, Abelardo Pardo, Jose A. Ruiperez-Valiente, David Arellano Martin-Caro, and Carlos Delgado Kloos. Detection and evaluation of emotions in massive open online courses. *Journal of Universal Computer Science*, 21(5):638–655, 2015.

[110] S. Owen, R. Anil, T. Dunning, and E. Friedman. *Mahout in action.* Manning Publications Co., 2011.

[111] ROLE Consortium. ROLE Project. http://www.role-project.eu/, 2009-2012. Last visited September 2012.

[112] Derick Leony, Abelardo Pardo, Luis de la Fuente Valentín, David Sánchez de Castro, and Carlos Delgado Kloos. Glass: A learning analytics visualization tool. In *Proceedings of the 2Nd International Conference on Learning Analytics and Knowledge*, LAK '12, pages 162–163, New York, NY, USA, 2012. ACM.

[113] Pedro J. Muñoz Merino and Carlos Delgado Kloos. An architecture for combining semantic web techniques with intelligent tutoring systems. In *Proceedings of the Intelligent Tutoring Systems conference (B. P. Woolf, E. Aïmeur, R. Nkambou, and S. P. Lajoie, eds.), vol. 5091 of Lecture Notes in Computer Science*, pages 540–550. Springer, 2008.

Appendix A

Script to detect emotions from student events

```
#!/usr/bin/python

# db module
# Derick Leony <dleony@it.uc3m.es>

import MySQLdb
import json

def connect_db(server, usr, pwd, dbn):
    con = None
    try:
        con = MySQLdb.connect(server, usr, pwd, dbn)
    except _mysql.Error, e:
        print "Error␣%d:␣%s" % (e.args[0], e.args[1])
    return con

def get_users(con):
    cur = con.cursor()
    query = ("SELECT␣r.id,␣r.entityId␣"
             "FROM␣relatedentity␣r,␣eventrelatedentity␣er␣"
             "WHERE␣r.id␣=␣er.relatedentityid␣"
             "AND␣er.role␣=␣'user'␣"
             "GROUP␣BY␣entityId,␣id␣"
             "ORDER␣BY␣entityId")
    cur.execute(query)
    users = cur.fetchall()
    return users
```

```
def get_real_emotions_of_user(con, user_id):
    cur = con.cursor()
    query = ("SELECT created_at, happyness, boredom, confusion, frustration "
             "FROM emotion "
             "WHERE user_id = %s "
             "AND frustration >= 50 "
             "ORDER BY created_at")
    cur.execute(query, (user_id))
    emotions = cur.fetchall()
    return emotions

def get_events_for_user_emotion(con, user_id, timestamp, window_size):
    cur = con.cursor()
    query = ("SELECT e.id, e.name, e.datetime "
             "FROM event e, eventrelatedentity er "
             "WHERE er.relatedentityid = %s "
             "AND er.role = 'user' "
             "AND er.eventid = e.id "
             "AND e.datetime between '2012-11-28 11:30' and %s "
             "ORDER BY datetime DESC, id "
             "LIMIT %s")
    cur.execute(query, (user_id, timestamp, window_size))
    events = list(cur.fetchall())
    events.reverse()
    return events

def get_users_from_sample(con, sample_id):
    cur = con.cursor()
    query = ("SELECT r.id, r.entityId "
             "FROM relatedentity r, eventrelatedentity er, sample_user su "
             "WHERE er.role = 'user' "
             "AND su.sample_id = %s "
             "AND r.id = er.relatedentityid "
             "AND r.entityId = su.nia "
             "GROUP BY entityId, id "
             "ORDER BY entityId")
    cur.execute(query, (sample_id))
    users = cur.fetchall()
    return users

def eval_event_message(con, event_id):
    cur = con.cursor()
    query = ("SELECT TRIM(BOTH '\"' FROM r.entityId) "
             "FROM eventrelatedentity er, relatedentity r "
             "WHERE er.eventid = %s "
             "AND er.role = 'messages' "
             "AND er.relatedentityid = r.id")
    cur.execute(query, (event_id))
```

```
    evaluation = 'ok'
    row = cur.fetchone()
    if row is not None:
        if row[0] != '':
            evaluation = 'error'
    return evaluation

def eval_resource(con, event_id):
    cur = con.cursor()
    query = ("SELECT r.entityId "
             "FROM eventrelatedentity er, relatedentity r "
             "WHERE er.eventid = %s "
             "AND er.role = 'invocation' "
             "AND er.relatedentityid = r.id")
    cur.execute(query, (event_id))
    evaluation = 'resource_internal'
    row = cur.fetchone()
    if row is not None:
        if row[0].startswith('http'):
            if 'google' in row[0]:
                evaluation = 'resource_search'
            elif 'uc3m.es' not in row[0]:
                evaluation = 'resource_external'
    return evaluation

def get_events(con, user_id):
    cur = con.cursor()
    query = ("SELECT e.id, e.name, e.datetime "
             "FROM event e, eventrelatedentity er "
             "WHERE er.relatedentityid = %s "
             "AND er.role = 'user' "
             "AND er.eventid = e.id "
             "AND e.datetime between '2012-11-28 11:30' and '2012-11-29 17:00' "
             "ORDER BY datetime, id")
    cur.execute(query, (user_id))
    events = cur.fetchall()
    return events

def get_prediction(con, script, model_id, window_size):
    q = ("SELECT prediction_id "
         "FROM prediction "
         "WHERE window_size = %s "
         "AND script = %s "
         "AND model_id = %s")
    cur = con.cursor()
    cur.execute(q, (window_size, script, model_id))
    prediction = cur.fetchone()
    if prediction is None:
```

```
        q = ("INSERT INTO prediction (window_size, script, model_id) "
             "VALUES (%s, %s, %s)")
        cur.execute(q, (window_size, script, model_id))
        prediction_id = cur.lastrowid
    else:
        prediction_id = prediction[0]
        q = ("DELETE FROM predicted_emotion "
             "WHERE prediction_id = %s ")
        cur.execute(q, (prediction_id))
    con.commit()
    cur.close()
    return prediction_id

def create_emotion(con, pred_id, user, dtime, emotions, queue_size):
    print pred_id, user, dtime, emotions, queue_size
    q = ("INSERT INTO predicted_emotion "
         "(prediction_id, user_id, created_at, happyness, boredom, confusion, "
         "frustration, length) "
         "VALUES (%s, %s, %s, %s, %s, %s, %s, %s)")
    cur = con.cursor()
    cur.execute(q, (pred_id, str(user), dtime) + tuple(emotions) + (queue_size, ))
    con.commit()
    cur.close()

def save_model(con, model):
    q = ("INSERT INTO model "
         "(emotion, initials, transitions, outputs) "
         "VALUES (%s, %s, %s, %s)")
    cur = con.cursor()
    cur.execute(q, (model.label, json.dumps(model.initial),
        json.dumps(model.transitions), json.dumps(model.productions)))
    con.commit()
    last_id = cur.lastrowid
    cur.close()
    return last_id

#!/usr/bin/python

# detectemo module
# Derick Leony <dleony@it.uc3m.es>

from collections import deque
import ghmm

class Observation:
    id = 0
    name = ''
    datetime = ''
```

```
    def __init__(self, id = 0, name = '', datetime = ''):
        self.id = id
        self.name = name
        self.datetime = datetime

    def setName(self, name):
        self.name = name

    def __repr__(self):
        return self.name

class Emotion:
    id = None
    label = ''
    model = None
    alphabet = None
    transitions = None
    productions = None
    initial = None

    def __init__(self, label, alphabet, transitions, productions, initial):
        self.label = label
        self.alphabet = ghmm.Alphabet(alphabet)
        self.transitions = transitions
        self.productions = productions
        self.initial = initial
        self.model = ghmm.HMMFromMatrices(self.alphabet, ghmm.DiscreteDistribution(self.alphabet),
                transitions, productions, initial)

    def evaluate(self, queue):
        sequence = ghmm.EmissionSequence(self.alphabet, list(queue))
        return self.model.loglikelihood(sequence)

    def train(self, queue):
        print self.alphabet
        seq = ghmm.EmissionSequence(self.alphabet, queue)
        self.model.baumWelch(seq)
        print self.model

    def multi_train(self, queues):
        seqs = ghmm.SequenceSet()
        for queue in queues:
            seqs.merge(ghmm.EmissionSequence(self.alphabet, list(queue)))

        self.model.baumWelch(seqs)
        print self.model
```

```
    def __repr__(self):
        return self.label

import numpy as np

def probs(length, trail=[]):
    limit = round(1.0 - sum(trail), 2)
    if length == 0:
        yield trail + [limit, ]
    else:
        step = 10
        norm = int(limit * 100) + 1
        for i in range(0, norm, step):
            for p in probs(length - 1, trail + [i / 100.0, ]):
                yield p

def combinator(generator, length, trail=[]):
    if length == 0:
        yield trail
    else:
        for el in generator:
            for gen_el in combinator(generator, length-1, trail+[el,]):
                yield gen_el

def delimited_probs(base, current=0, trail=[]):
    limit = round(1.0 - sum(trail), 2)
    if current == (len(base) - 1):
        yield trail + [limit, ]
    else:
        # freedom = 3
        step = 10
        norm = int(limit * 100) + 1
        prob = base[current][0]
        freedom = base[current][1]
        if freedom == 0:
            n_range = [prob]
        else:
            n_range = range( max(0, prob - freedom * step),
                min(prob + freedom * step, norm), step)
        for i in n_range:
            for p in delimited_probs(base, current + 1, trail + [i / 100.0, ]):
                yield p

def matrix_delimited_probs(matrix, current=0, trail=[]):
    if current == len(matrix):
        yield trail
    else:
        for el in delimited_probs(matrix[current]):
```

```
            for gen_el in matrix_delimited_probs(matrix, current + 1, trail+[el,]):
                yield gen_el

#!/usr/bin/python

# Main script
# Derick Leony <dleony@it.uc3m.es>

import db
import argparse
from collections import deque
from ghmm import *
import detectemo
import probs

def normalize_observation(con, obs):
    # 1. Check for events to omit: lms_choice_*, lms_user_*, lms_upload_*
    # 2. Check for specific patterns to summarize: lms_course_*, lms_forum_*, lms_discussion_*
    # 3. Check for events that bifurcate: gcc, valgrind, visit_url
    # 4. Everything else is left unchanged: bashcmd, gdb, ide, text_editor

    if obs.name.startswith('lms_choice_') or \
       obs.name.startswith('lms_user_') or \
       obs.name.startswith('lms_upload_'):
        obs = None
    elif obs.name.startswith('lms_course_'):
        obs.name = 'lms'
    elif obs.name.startswith('lms_forum_') or \
         obs.name.startswith('lms_discussion_'):
        obs.name = 'forum'
    elif obs.name.startswith('lms_resource_'):
        obs.name = 'resource_internal'
    elif obs.name == 'gcc' or \
         obs.name == 'valgrind':
        obs.name += '-' + db.eval_event_message(con, obs.id)
    elif obs.name == 'visit_url':
        obs.name = db.eval_resource(con, obs.id)

    return obs

def get_observations(con, user_id):
    events = db.get_events(con, user_id)
    observations = []
    for event in events:
        obs = detectemo.Observation(event[0], event[1], event[2])
        obs = normalize_observation(con, obs)
        if obs is not None:
            observations.append(obs)
```

```
    return observations

def process_events(con, script_id, model, window_size):
    pred_id = db.get_prediction(con, script_id, model.id, window_size)
    users = db.get_users_from_sample(con, 1)
    for counter, user in enumerate(users):
        print "user:%s (%d/%d)" % (user[1], counter + 1, len(users))
        observations = get_observations(con, user[0])
        queue = deque()
        generated = False
        personal = [50] * 4
        for obs in observations:
            queue.append(obs.name)
            if len(queue) > window_size:
                queue.popleft()
            if len(queue) < window_size and generated:
                continue
            generated = True
            max_prob = 0
            max_emo = None
            probs = []
            prob = model.evaluate(queue)

            if (prob == -float('Inf')):
                prob = 0
            db.create_emotion(con, pred_id, user[0], obs.datetime, [prob, 0, 0, 0],
                              len(queue))
            print "%s, %s, %s, %s \n" % (obs.datetime, prob, obs.id, obs.name)

def define_emotions(con, script_id, window_size):
    alphabet = ['bashcmd', 'forum', 'gcc-error', 'gcc-ok', 'gdb',
        'ide', 'lms', 'text_editor', 'resource_external', 'resource_internal',
        'resource_search', 'valgrind-error', 'valgrind-ok']

    # same order as in alphabet list
    productions = [
            [ .25, .10, .15, .10, .05, .05, .10, .10, .00, .00, .00, .05, .05], # work
            [ .05, .00, .70,  00, .00, .00, .00, .00, .00, .00, .00, .25, .00], # problem
            [ .05, .00, .00, .50, .05, .05, .00, .00, .00, .00, .00, .00, .35], # solve
            [ .00, .30, .00, .00, .00, .00, .10, .00, .10, .30, .20, .00, .00], # look
            [ .00, .00, .00, .00, .00, .00, .05, .00, .95, .00, .00, .00, .00]  # distract
            ]

    initial = [1.0 / len(productions)] * len(productions)

    # emotions labels
    labels = ['happiness', 'confusion', 'frustration', 'boredom']
```

```
transitions = []

# happiness
transitions.append([
    [[60, 1], [10, 0], [10, 0], [10, 0], [10, 0]],
    [[30, 1], [20, 1], [10, 0], [30, 1], [10, 0]],
    [[60, 1], [10, 0], [10, 0], [10, 0], [10, 0]],
    [[40, 1], [10, 0], [20, 1], [20, 1], [10, 0]],
    [[60, 1], [10, 0], [10, 0], [10, 0], [10, 0]],
    ])

# confusion
transitions.append([
    [ 0.4, 0.3, 0.05, 0.05,  0.2],
    [0.15, 0.5, 0.05, 0.15, 0.15],
    [ 0.6, 0.2, 0.05, 0.05,  0.1],
    [ 0.2, 0.3,  0.1,  0.2,  0.2],
    [ 0.2, 0.3, 0.05, 0.05,  0.4],
    ])

# frustration
transitions.append([
    [ 0.4,  0.3,  0.1,  0.1,  0.1],
    [ 0.1,  0.5, 0.05,  0.2, 0.15],
    [ 0.7, 0.15, 0.05, 0.05, 0.05],
    [0.25,  0.5, 0.05,  0.1,  0.1],
    [ 0.3,  0.3, 0.05, 0.15,  0.2],
    ])

# boredom
transitions.append([
    [ 0.4,  0.1,  0.1,  0.1, 0.3],
    [ 0.1,  0.2,  0.1,  0.2, 0.4],
    [ 0.4,  0.1, 0.05, 0.05, 0.4],
    [0.15,  0.2, 0.15,  0.2, 0.3],
    [ 0.2,  0.1, 0.05, 0.05, 0.6],
    ])

# emotions = []
# gen = list(probs.probs(4))
for trans in probs.matrix_delimited_probs(transitions[0]):
    print "Generating happiness for ", trans
    model = detectemo.Emotion("happiness", alphabet, trans, productions,
            initial)
    model.id = db.save_model(con, model)
    process_events(con, script_id, model, window_size)

return emotions
```

```
def main():
    script_id = 7
    parser = argparse.ArgumentParser(description='Script to infer emotions from CAM database.')
    parser.add_argument('window_size', type=int, metavar='W',
                        help='Amount of events to take into account')
    args = parser.parse_args()
    con = db.connect_db('localhost', 'affective', 'affectivelearning',
          'as_mockup_2012')
    if con:
        emotions = define_emotions(con, script_id, args.window_size)
        con.close()

if __name__ == "__main__":
    main()
```

Appendix B

Questionnaires used in user study 2 of visualization evaluation

Questionaire to collect data

The first step is to create an anonymous ID for your responses: Provide a sequence of 4 letters followed by a sequence of 4 numbers. Make sure to write this down in order to use the same ID in further surveys and to be able to see the data in the visualizations. For example, you can use "CASE1571" or "HOME4865". Please do not use obvious sequences like "AAAA0000" or "ABCD1234".
Short text answer

Please indicate your gender:

- *Male*
- *Female*

Please indicate your age range:
Short text answer

Please indicate the week number for which you are reporting data:
Short text answer

Affective questions

Indicate how frequently you have felt FRUSTRATED while working on the following activities
If you have not yet worked on an activity (e.g. creation of installation instructions),

then please select ”never” for this activity.

	Never	Rarely	Occasionally	A moderate amount	A great deal
Brainstorming					
Interface design					
Implementation					
Creation of documentation					
Experiments with users					
Creation of installation instructions					

Indicate how frequently you have felt CONFUSED while working on the following activities
If you have not yet worked on an activity (e.g. creation of installation instructions), then please select ”never” for this activity.

	Never	Rarely	Occasionally	A moderate amount	A great deal
Brainstorming					
Interface design					
Implementation					
Creation of documentation					
Experiments with users					
Creation of installation instructions					

Indicate how frequently you have felt BORED while working on the following activities
If you have not yet worked on an activity (e.g. creation of installation instructions), then please select ”never” for this activity.

	Never	Rarely	Occasionally	A moderate amount	A great deal
Brainstorming					
Interface design					
Implementation					
Creation of documentation					
Experiments with users					
Creation of installation instructions					

Indicate how frequently you have felt HAPPY while working on the following activities

If you have not yet worked on an activity (e.g. creation of installation instructions), then please select ”never” for this activity.

	Never	Rarely	Occasionally	A moderate amount	A great deal
Brainstorming					
Interface design					
Implementation					
Creation of documentation					
Experiments with users					
Creation of installation instructions					

Indicate how frequently you have felt MOTIVATED while working on the following activities.

If you have not yet worked on an activity (e.g. creation of installation instructions), then please select ”never” for this activity.

	Never	Rarely	Occasionally	A moderate amount	A great deal
Brainstorming					
Interface design					
Implementation					
Creation of documentation					
Experiments with users					
Creation of installation instructions					

Time dedication

Indicate an approximate number of hours that you have dedicated to the course this week.

- <2 hours
- 2-4 hours

- 4-6 hours
- 6-8 hours
- 8-10 hours
- >10 hours

Questionaire to evaluate visualizations

Evaluation emotion and time visualization

Please enter the same ID you used in the data collection form:

Short text answer

Do you find the different charts useful?

	1 - not useful at all	2	3 - on average	4	5 - very useful
Emotion chart					
Time line chart					
Heat map					
Scatter plot					

Are you below or above average compared to the other students?

	1 - below average	2	3 - on average	4	5 - above average
Frustration					
Confusion					
Boredom					
Happiness					
Motivation					
Time dedication					

My most frequent emotion during the class has been:

- Frustration
- Confusion
- Boredom
- Happiness
- Motivation

By using the Heat map, can you find another student who is similar to you? If yes, provide his or her ID (e.g. Student 32).

Short text answer

The activity that has motivated students the most has been

- Brainstorming
- Interface design
- Implementation
- Creation of documentation
- Experiment with users
- Creation of installation instructions
- No activity has motivated students

The activity that has frustrated me the most has been

- Brainstorming
- Interface design
- Implementation
- Creation of documentation
- Experiment with users
- Creation of installation instructions
- No activity has frustrated me

The activity where I am most different from other students is

- Brainstorming
- Interface design
- Implementation
- Creation of documentation

- Experiment with users
- Creation of installation instructions
- No activity has frustrated me

What other data would you like to have visualized or made accessible? Give a priority according to your need.

	1 - lowest priority	2	3 - medium	4	5 - highest priority
used resource					
student information					
comparing students					
detailed statistics					
Motivation					
Time dedication					

Would you like to continue using the tool?

- Yes
- No
- I don't know

I think I would like to use this tool frequently.

	1	2	3	4	5	
Strongrly agree						Strongly disagree

I found the tool unnecessary complex.

	1	2	3	4	5	
Strongrly agree						Strongly disagree

I thought the tool was easy to use.

I think that I would need the support of a technical person to be able to use this tool.

I found the various functions in the tool were well integrated.

I thought there was too much inconsistency in this tool.

I imagine that most people would learn to use this tool very quickly.

	1	2	3	4	5	
Strongrly agree						Strongly disagree

	1	2	3	4	5	
Strongrly agree						Strongly disagree

I found the tool very awkward to use (”awkward” means ”lacking ease or grace”).

I felt very confident using the tool.

I needed to learn a lot of things before I could get going with this tool.

How likely are you going to recommend this tool to a friend or colleague?

Strongrly agree	1	2	3	4	5	Strongly disagree

Strongrly agree	1	2	3	4	5	Strongly disagree

Strongrly agree	1	2	3	4	5	Strongly disagree

Strongrly agree	1	2	3	4	5	Strongly disagree

Strongrly agree	1	2	3	4	5	Strongly disagree

Strongrly agree	1	2	3	4	5	Strongly disagree

Very unlikely	1	2	3	4	5	Very likely