An Affect-Aware Intelligent Tutoring System for EmoRegulators

A Restricted-Perception Wizard-of-Oz Approach

Pedro Marques do Carmo Rodrigues

Thesis to obtain the Master of Science Degree in Information Systems and Computer Engineering

Supervisors: Prof. João Miguel De Sousa de Assis Dias
Prof. Ana Maria Severino de Almeida e Paiva

Examination Committee

Chairperson: Prof. Daniel Jorge Viegas Gonçalves
Supervisor: Prof. João Miguel De Sousa de Assis Dias
Members of the Committee: Prof. Rui Filipe Fernandes Prada

June 2018
Acknowledgments

I want to express my sincerest thank you to everyone in my family; especially to my mother and my grandparents. For their unceasing support and relentless availability. For their valuable lessons and guidance. For always pushing me to do better and to know more - this dissertation is especially dedicated to you.

To my friends, for motivating me and keeping me in good spirits. And for always making me aware of the bigger picture.

I would also like to acknowledge my dissertation supervisors Prof. João Dias and Prof. Ana Paiva for sharing the knowledge that has made this Thesis possible; especially to Prof. João Dias, for always showing patience and kindness.

To Professor Esther J. Schek, for her availability to take part in the tests that were conducted.

Thank you all for the support.
Abstract

This project addresses the problem of how to create an intelligent, adaptive system capable of helping to regulate the players’ affective states during an interaction session in EmoRegulators, a serious game that teaches players to be aware of their emotions and to regulate them. We conducted research into other affect-aware Intelligent Tutoring Systems to learn which emotion regulation strategies they use. Unlike many of these emotion-aware ITS which use a set of rules to activate the usage of emotion regulation strategies, we propose a model heavily based on restricted-perception Wizard-of-Oz studies, where a human expert plays the role of a tutor during a demonstration phase by controlling the facilitator (an instructional agent) in EmoRegulators, using an interface which was carefully designed to enforce that the expert has the same perceptions as the agent. We created a set of features that represent the game state and the affective state that could be used later in the training phase, using the data collected from users in the WoZ demonstrations. This data was pre-processed and fed to a selection of machine learning algorithms to study their performance in EmoRegulators specifically. Users reported an overall positive experience during the demonstrations but model performance didn’t meet our initial expectations; after a merging and grouping of strategies was applied, though, we saw a slight increase in recall score at a fairly reasonable compromise in accuracy for the Leave-One-Subject-Out approach.

Keywords

Emotion Regulation; Intelligent Tutoring Systems; Serious Games; EmoRegulators; Wizard-of-Oz Studies; Machine Learning
Resumo

Este projecto aborda o problema de como criar um sistema adaptável, inteligente e capaz de ajudar a regular os estados afectivos de jogadores durante uma sessão interactiva no EmoRegulators, um jogo sério que ensina os jogadores a estarem conscientes das suas emoções e a regulá-las. Foi feita investigação sobre sistemas de tutoria inteligentes afectivos para aprender que estratégias de regulação de emoções usam. Ao contrário de muitos sistemas de tutoria inteligentes, que usam conjuntos de regras para activar as estratégias de regulação de emoções, nós propomos um modelo fortemente baseado no conceito de estudos Wizard-of-Oz (WoZ) com restrição de percepções, em que um especialista humano faz o papel de tutor durante uma fase de demonstrações ao controlar a facilitadora (um agente instrutivo) no EmoRegulators, através de uma interface que foi cuidadosamente desenhada para forçar o especialista a ter as mesmas percepções que o agente. Criámos um conjunto de características que representam o estado do jogo e o estado afectivo e que iriam a ser usados mais tarde na fase de treino, usando os dados recolhidos aos utilizadores durante as demonstrações WoZ. Estes dados foram pré-processados e alimentados a uma selecção de algoritmos de aprendizagem automática para estudar o seu desempenho especificamente no EmoRegulators. Os utilizadores reportaram uma experiência em geral positiva nas demonstrações mas os nossos resultados na escolha de modelos ficou aquém das nossas expectativas iniciais; no entanto, após aplicarmos uma estratégia de agrupamento das interacções, verificámos algumas melhorias em recall perante um compromisso razoável em accuracy para a abordagem Leave-One-Subject-Out.

Palavras Chave

Regulação de Emoções; Sistemas Tutoriais Inteligentes; Jogos Sérios; EmoRegulators; Estudos Wizard-of-Oz; Aprendizagem Automática
## Contents

1 Introduction .................................................. 1
   1.1 Motivation .................................................. 3
   1.2 Problem ................................................... 4

2 Theory and Background ........................................ 7
   2.1 Emotion Regulation ........................................ 9
      2.1.1 Gross's Construct of Emotion Regulation .......... 9
      2.1.2 Other Views on Emotion Regulation .............. 10
         2.1.2.A Attribution Theory ............................ 10
         2.1.2.B Cognitive Disequilibrium Theory .......... 10
         2.1.2.C Cognitive Reappraisal .................... 11
   2.2 Machine Learning ......................................... 11
      2.2.1 Machine Learning Algorithms .................... 11
         2.2.1.A Gaussian Naive Bayes ...................... 11
         2.2.1.B k-Nearest Neighbors ...................... 12
         2.2.1.C Decision Tree ............................... 12
         2.2.1.D Random Forest .............................. 13
         2.2.1.E Multilayer Perceptron (MLP) .......... 14
      2.2.2 Performance Estimation ........................... 15
      2.2.3 Underfitting and Overfitting .................... 17

3 Related Work ................................................ 21
   3.1 Emotion regulation in ITS ............................... 23
      3.1.1 AutoTutor and Affective AutoTutor ............ 23
      3.1.2 EER-Tutor ........................................... 24
         3.1.2.A Identifying Users’ Affective States ........ 24
         3.1.2.B Affective Pedagogical Agent for EER-Tutor .. 25
         3.1.2.C Experiment and Results .................... 26
   3.1.3 A web-based learning system ....................... 26
3.2 Emotion Regulation in Serious Health Games ........................................ 27
  3.2.1 EmoRegulators ................................................................. 27
  3.2.2 PlayMancer ................................................................. 30
3.3 Wizard-of-Oz (WoZ) Studies ................................................................. 31
  3.3.1 Learning from the Wizard (LW) ........................................... 31
  3.3.2 Restricted-Perception WoZ Studies ...................................... 33
    3.3.2.A Data Collection ...................................................... 33
    3.3.2.B Strategy Extraction ................................................ 34
    3.3.2.C Strategy Refinement ............................................... 35
    3.3.2.D Practical Use and Results ...................................... 35
4 Solution .................................................. 37
  4.1 Emotion Regulation: Defining Appropriate Strategies for EmoRegulators ........ 39
  4.2 Activation of the Emotion Regulation Strategies .................................. 41
    4.2.1 Methodology ............................................................. 42
    4.2.2 Game State and Affective State Representation ...................... 43
      4.2.2.A Data Collection .................................................. 46
    4.2.3 Restricted-Perception WoZ Interface ................................ 48
      4.2.3.A Design decisions ................................................ 50
    4.2.4 Data Preparation .......................................................... 51
      4.2.4.A Oversampling ...................................................... 52
    4.2.5 Training Phase .............................................................. 54
      4.2.5.A k-Fold Cross Validation ........................................... 55
      4.2.5.B Leave-One-Subject-Out ........................................ 56
      4.2.5.C Leave-One-Exercise-Out ........................................ 57
    4.2.6 Runtime Classification .................................................. 58
5 Results .............................................. 61
  5.1 Restricted-perception WoZ demonstrations ...................................... 63
    5.1.1 WoZ Protocol .......................................................... 64
    5.1.2 Participants' Feedback .............................................. 65
  5.2 Model Selection ................................................................. 66
6 Conclusion .............................................. 73
  6.1 Conclusions ................................................................. 75
  6.2 System Limitations and Future Work ........................................... 76
<table>
<thead>
<tr>
<th></th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Interface Icon Matching Questionnaire</td>
<td>83</td>
</tr>
<tr>
<td>B</td>
<td>WoZ Demonstration Consent Form</td>
<td>89</td>
</tr>
</tbody>
</table>
Introduction

Contents

1.1 Motivation ................................................................. 3
1.2 Problem ................................................................. 4
1.1 Motivation

Researchers have been attempting to use intelligent autonomous agents for educational purposes [1]. Their goal is to replicate a real teacher’s success as best as possible in the form of tutoring systems or pedagogical agents, which intend to teach students a particular subject or method, or to perform specific tasks.

According to Malekzadeh et al., an "Intelligent Tutoring System (ITS) is a computer-based educational system that provides individualized instructions similar to like a human tutor. Typical ITSs determine how and what to teach a student based on the learner’s pedagogical state to enhance learning." [2].

Additionally to considering the learner’s pedagogical state it is equally important to address the learner’s emotions. As humans we resort to emotions to guide our thinking process all the time. Ahn and Picard have recently found that emotions affect several cognitive and behavioral processes, such as decision-making [3]. Furthermore, the social cognitive theory tells us that learning is highly impacted by the affection-cognition correlation [4].

By now we all know how most times it is the methodology of a teacher that separates the great ones from the not-so-good ones. Not only that, but it is also their emotional intelligence and their ability to maintain good interpersonal relationships with their students, that often dictates the latter’s overall success in learning [5].

To better understand the concept of emotional intelligence we can refer to Salovey and Mayer [6] who consider emotional intelligence as a subset of social intelligence that involves assessing one’s own and others’ emotions, as well as identifying them, and using that information to guide one’s thinking and actions.

The way we manage emotions is also very important, especially in the context of relationships. Our ability to control which emotions we express, how intense they are and for how long they last can dictate how beneficial or harmful they can be in a given situation [7]. This is only a fraction of what is called emotion regulation, and Gross defines it as "the heterogeneous set of processes by which emotions themselves are regulated" (a more in-depth look at emotion regulation can be found in the following section). Managing our and others’ emotions in such a way is also one of the four core skills that comprise emotional intelligence according to Mayer et al. [8].

Given the relevance of emotions to behavior, it should come as no surprise that researchers have already attempted to explore emotion-aware ITS in hopes to replicate the success of real teachers. D’Mello and Graesser’s Affective AutoTutor [9], for example, is able to formulate empathetic replies to negative emotional states such as frustration and boredom, as well as attempting to help a student recover from a state of confusion when he/she doesn’t understand something - a form of emotion regula-
tion in itself. However, these affective capabilities didn’t have different results compared to their regular AutoTutor. Despite this particular example, in his review of emotion regulation in ITS [2] Malekzadeh et al. concluded that emotion-aware ITSs have better results than their non-emotional counterparts, namely achieving more positive emotions [10, 11] and satisfaction and positive impressions [12] in the learners.

1.2 Problem

The above cited examples achieve emotion regulation in the user by monitoring facial-features, speech contours, body language, interaction logs, language and peripheral physiology, to name a few [13]. Focusing on the latter, The EmoRegulators [14], a serious game for learning how to regulate emotions, attempts to teach humans to regulate theirs through a series of BEAR-based (see [15] for the original BEAR) sessions composing a game, by measuring physiological and biological data such as heart rate and electrodermal activity (EDA). Despite its ability to teach these emotion regulation mechanisms to the player, the EmoRegulators lacks the adaptability and the autonomy that ITSs should pack to thrive. The system isn’t capable of acting upon stressful or frustrating emotions that the player might feel while performing the proposed sessions. This poses the following problem:

“How to create an intelligent, adaptive system in EmoRegulators, capable of helping to regulate players’ affective states during an interaction session?”

Several affect-aware ITS at the time of writing use rule-based systems to activate the usage of emotion regulation strategies after assessing a user’s emotional state. Auto-tutor, for example, does this by mapping dynamic assessments of the students’ cognitive and affective states with appropriate tutor actions [9].

On another note, teachers are also responsible for teaching classes that often have students with different learning curves and cognitive abilities; good teachers can guide both types of students at the same time with optimal efficacy and help them achieve good results, which is something ITSs have struggled with in the past due to expertise-reversal literature, which basically states that methods which promote learning for low-knowledge students can impair the knowledge of high-knowledge students [16]. Also noteworthy to achieve good results, and as corroborated by Moreno’s conclusion, is the fact that the most successful key point of pedagogical agents - which ITSs are an example of - lies in the specific instructional method embedded in the agent [17].

This idea that meeting a student’s specific learning needs in pedagogical environments is key, coupled with the aforementioned conclusions regarding the efficacy of emotion-aware ITS when compared to their non-emotional counterparts, brings us to the hypothesis we propose in this paper:
"The usage of emotion regulation strategies based on expert knowledge from human tutors will help emotion-aware intelligent tutoring systems better replicate their ability to understand and regulate users’ emotions."

In the following chapters we delve into the concepts and state-of-the-art involved in ITS and serious games. The next section, Theory and Background, introduces some emotion regulation strategies that have been used by researchers in ITS. Then we take a look at the state-of-the-art when it comes to emotion regulation in ITS, emotion regulation in serious health games - including a detailed look at EmoRegulators - and work based on a well-known learning framework that is used to teach expert knowledge and methods to agents. In the fourth section, we describe our solution for the identified problem and then discuss results and draw conclusions regarding the conducted work.
Theory and Background

Contents

2.1 Emotion Regulation ..................................................... 9
2.2 Machine Learning ...................................................... 11
2.1 Emotion Regulation

This section describes key concepts that are important to understand in order to address the problems of Emotion Regulation and Intelligent Tutoring Systems. First we discuss Gross’s classification of emotion regulation strategies; then we look specifically at some strategies such as the Attribution Theory, the Cognitive Disequilibrium Theory and the concept of Cognitive Reappraisal.

2.1.1 Gross’s Construct of Emotion Regulation

As we’ve seen before, Gross defines emotion regulation (in the sense of the regulation of emotions) as a diverse set of processes that regulate emotions, simply put. He also draws an important parallel between intrinsic regulation, which corresponds to one’s regulation of his own emotions (self-regulation), and extrinsic regulation, corresponding to one’s regulation of others’ emotions (interpersonal emotion regulation).

In 2001 [18] Gross proposed a process model of emotion regulation that divides emotion regulation strategies into five families according to the point in time when they have their primary impact on the emotion-generative process, which can be seen in Figure 2.1.

Situation Selection

Situation Modification

Attentional Deployment

Cognitive Change

Response Modulation

Figure 2.1: The five families of emotion regulation strategies according to Gross and Thomson (2007).

In 2001 [18] Gross proposed a process model of emotion regulation that divides emotion regulation strategies into five families according to the point in time when they have their primary impact on the emotion-generative process, which can be seen in Figure 2.1.

Situation selection is the first type of emotion regulation and happens in the earliest possible moment. It corresponds to taking actions that make it more likely for one to be in a situation that enables emotions we desire (or taking actions that make it less likely for one to end up in a situation that enables emotions we don’t desire). A second form of performing emotion regulation consists of attempting to modify the situation directly to change its emotional effect. An example of this is one suggesting his/her family to play a board game in the case of a power outage that makes it impossible to watch a TV show, instead of allowing for negative emotions to take place.

While the previous two forms of emotion regulation involve changing the environment, the following
Attentional deployment comes in two types: distraction and rumination. The former involves a shift of attention away from the emotional aspects of a situation or away from the situation in its entirety. The latter involves a sustained focus on thoughts and feelings bound to an emotion-eliciting situation. Cognitive change corresponds to changing one or more emotion generation processes (appraisals) in a way that alters the situation's emotional meaning, by internally changing one's interpretation of the situation.

Lastly, response modulation happens after response tendencies have started. It refers to affecting physiological, experiential or behavioral responses relatively directly. An example would be one's decision to control his breathing and counting to ten to control one's anger.

### 2.1.2 Other Views on Emotion Regulation

In addition to Gross, other researchers have studied emotion regulation strategies from different perspectives. These concepts are hereby introduced so that the reader can easily understand certain aspects mentioned in the Related Work.

#### 2.1.2.A Attribution Theory

While there are many attribution “theories”, the common ideas are that people interpret behavior in terms of its causes and that these interpretations play an important role in determining reactions to the behavior [19]. In the particular case of AutoTutor, for example, it attempts to attribute the cause of a certain situation, such as the student feeling bored, to a particular factor, entity or event, such as the learning content being uninteresting, the tutor itself or the student.

#### 2.1.2.B Cognitive Disequilibrium Theory

In his theory of cognitive development, Jean Piaget describes cognitive disequilibrium as a state of cognitive imbalance [20]. One experiences such a state of imbalance when acquiring information that requires us to develop a new schema (a building block of knowledge) or modify an existing one.

Disequilibrium is an uncomfortable state for a person, and as such, we attempt to return to a state of equilibrium as quickly as possible. If there is something about our environment that doesn’t fit our existing schema, we will either dedicate mental energy to develop a new one, or adapting an existing schema. For example, a child learning how to tie their shoes has to physically maneuver the shoe laces while thinking through the steps as they try to develop a new schema for shoe tying.
2.1.2.C Cognitive Reappraisal

Cognitive reappraisal is the process by which a person changes his/her interpretation of a certain emotional response by reinterpreting the meaning of the emotional stimulus [21]. This is a particular form of emotion regulation similar to Gross's cognitive change. For example, a student might be dissatisfied, sad even, with the results of an exam upon receiving them. Later on, the student might revisit this situation but look at it differently, reinterpreting the results as a way to challenge and improve oneself, and doing better next time or taking it as a valuable lesson. This emotion regulation strategy is divided into two moments:

i. Recognizing one’s negative response
ii. Reinterpreting the situation to suppress or overcome the severity of that response

This process can be very important in achieving a better emotional state, namely in a learning environment, and could perhaps be a useful coping strategy to implement in intelligent tutoring systems.

2.2 Machine Learning

This is an introductory section to some of the concepts that are later referenced in the solution. We present five machine learning algorithms and we discuss performance metrics and the problems of underfitting and overfitting.

2.2.1 Machine Learning Algorithms

2.2.1.A Gaussian Naive Bayes

In machine learning, naive Bayes classifiers are a set of supervised learning algorithms that apply the Bayes’ theorem with the naive assumption that there is independence between every pair of features.

Where the different members of this family of Bayes classifiers differ is in the distribution of the likelihood of the features. In the Gaussian Naive Bayes this distribution is assumed to be, as the name suggests, a Gaussian (or normal, as it is more widely known) distribution. These classifiers have worked quite well in real-world situations such as spam-filtering and they also only require a small amount of training data to estimate the necessary parameters [22]. In addition, they can be extremely fast compared to more sophisticated ones [23]. This algorithm also carries the advantage of supporting multi-class classification.

An issue with Naive Bayes classifiers, however, is that they aren’t good estimators. What this means is that if there are no occurrences of a class and a particular value for a feature together, then the
frequency-based probability estimate for this will be zero. This can be a problem as it results in an incorrect probability estimate since according to this method’s naive assumption of independence between features, the multiplication of all probabilities will also be zero. Nevertheless, while this makes it hard to explore the interactions between features (which could definitely be interesting as future work), this is often not a deal-breaker for when one just wants to do a classification task.

2.2.1.B k-Nearest Neighbors

Another classic method often used for classification and regression is kNN, short for k-Nearest Neighbors. The idea behind nearest neighbor methods is to find a predefined number of training samples closest to the new point distance-wise, and to then predict the label from these. This predefined number of samples is often a constant defined by the user (the k variable in the algorithm name), but it can also be based on a radius, meaning the number of samples varies according to the local density of points; that would instead be called radius-based neighbor learning, but here we focus on the former.

Although very simple - just as Naive Bayes explained previously - this class of algorithms has found plenty of success in numerous classification and regression problems such as handwritten text [24] and digit [25] recognition, and tissue classification based on gene expression [26].

One decision that has to be made when implementing this classifier is the choice of the distance function; while a custom function may be chosen based on particularities of a given problem, the Minkowski distance is often used, since it is a generalization of both the Euclidean distance and the Manhattan distance (two other popular choices for the distance function of the algorithm) and is among the highest scoring distance metrics for classification regarding accuracy [27, 28].

2.2.1.C Decision Tree

Just like kNN, Decision Trees (DTs) are a popular non-parametric supervised learning method used for classification and regression. They are called as such because of how problems can be represented according to a tree structure, where observations about an item are represented in the branches and the decisions, according to the ramifications of those observations, are represented as the tree’s leaves.

This is a very popular classifier due to how easy it is to understand, interpret and visualize its concept, as well as the fact that it uses a white box model: as long as a given scenario is observable in a model, one can easily explain each condition in that model by just using simple boolean logic, which can’t be said of other classifiers such as neural networks, where this is often extremely difficult to do. It also requires little data treatment, while other models often require data normalization, for example. There are many other advantages in using decision trees, but to mention a few others which we value the most, is the fact that it is computationally inexpensive to create decision trees even for large training
sets, and that these classifiers tend to handle data noise or irregularities pretty well, which makes for a well-rounded and robust choice.

Everything comes at a price, though. Decision trees are also well-known for tending to overfit if one isn’t careful. Fortunately, one can mitigate the likelihood of this occurring by tweaking the maximum depth of the tree or the number of samples at a leaf node. Also accounting for the possibility that some classes may dominate others in the data set - which could create trees that are biased towards those classes -, one can additionally balance the data set by normalizing the sum of the sample weights for each class to the same value.

2.2.1.D Random Forest

In machine learning, ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone [29–31]. One can usually consider two distinct types of ensemble methods:

- Averaging methods, where estimators are independently built and then their predictions are averaged. On average, the meta-estimator is usually better than any of its single constituents because its variance is reduced.

- Boosting methods, in which the base estimators are built sequentially and the purpose is to reduce the meta-estimator’s bias. The idea is to attempt to create a good performing ensemble by combining a number of weak models.

A popular ensemble method that falls in the former category and which is used in classification tasks is called random forest (or random decision forest). In this method, several decision trees are constructed at training time and the output is the class that is the mode of the classes, that is, the one that appears the most often. This mode is also called a majority vote, where each decision tree effectively votes for a particular class - its output. Although this is the case in the original publication by L. Breiman et al. [32], one can also combine the estimators by averaging their probabilistic prediction instead.

Given how the random forest classifier is built on top of several decision trees, it makes sense to look at the advantages and disadvantages of one compared to the other. Regarding disadvantages, there are a few downsides; it is more complex than decision trees and typically difficult to implement. It is also computationally expensive - at least compared to decision trees - and while it doesn’t actually follow a black model, it is fairly difficult to visualize the model or to understand the reasoning behind its predictions.

On the other hand, random forests overcome several problems with decision trees. The most relevant is, again, overfitting. By averaging several trees, though, there is a significantly lower risk of overfitting
in a random forest. Additionally, there is less variance and bias (we look at these two concepts later in this chapter), meaning one is less likely to find a classifier that doesn’t perform well based on how the train and test data relate. This is generally due to the bagging (also known as bootstrap aggregation) algorithm used in random forests [33].

2.2.1.E Multilayer Perceptron (MLP)

Artificial neural networks (ANNs) have seen a remarkable growth in interest and popularity in recent years. They are inspired by our brain, the most complex computing system we have knowledge of, which consists of a significantly large number of neurons. These powerful fundamental units of our brain work together, highly interconnected to each other, performing incredibly complex tasks. Artificial neural networks take this idea of building networks of single processing units, the neurons (which have in fact been studied as input/output devices [34]), to solve problems in numerous fields including data science and machine learning.

One of the algorithms that implement a neural network paradigm is called multilayer perceptron (MLP), originally introduced by Rumelhart et al. [35] in 1986. These networks consist of (1) a set of sensory units, also called the source nodes, which constitute the input layer, (2) one or more hidden layers of computation nodes, and (3) an additional final layer of computation nodes called the output layer (see Figure 2.2 [36]). The input signal is propagated through the multiple layers in the network in a forward direction, which is why these algorithms are also known as multilayer feedforward networks.

![Figure 2.2: A visual representation of the multiple layers that comprise a Multilayer Perceptron. Taken from Simon Haykin's "Neural Networks: A comprehensive foundation.". 2004](image-url)
Multilayer perceptrons and neural networks have been successfully implemented across different fields. Valery Petrushin used backpropagation neural networks to recognize emotions in speech in call centers [37] with an accuracy of 77%. S. Walter et al. performed a Wizard-of-Oz experiment to study the emotional behavior of test subjects, inducing particular emotions using techniques such as introducing delays and ignoring commands in a human-computer interaction scenario; several biometric sensors were used to collect bio-physiological data and, after improvements, achieved an accuracy of 72%. Sherif Yacoub [38] also used neural networks in a study of emotion recognition and emotion distinction from speech signals and achieved accuracy results upwards of 82% for certain emotion distinctions.

In multilayer perceptrons, the first step is to generate a mapping between input and output from training the network using a set of paired data. The edges between neurons of different layers, called weights, are then fixed and the network can be used to classify new data. As stated above, during classification, the signals are propagated from the input units across the net to find the activation values of the output units. These activation values are what determine - with the help of activation functions that use these values - the activation value of units in subsequent layers of the network; the activation functions of each unit sum together the contributions of all sending units i, where the contribution $Y$ of a unit $u$ is defined as

$$Y_u = \sum (weight_i \times input_i)$$ (2.1)

The activation functions usually add complexity to the above formula in several ways, such as normalizing and scaling the value to a 0 to 1 range or defining thresholds for neuron activation, for example.

### 2.2.2 Performance Estimation

There are several indicators when it comes to evaluating performance of learning algorithms. The most common ones are accuracy, precision, recall and F-score. When learning algorithms carry out prediction tasks, one way to assess their success is to express it in terms of the outcomes of what is called a confusion matrix (see Figure 2.3):

i. true positives (TP), also called hits, which refer to predicting a positive condition when the condition is indeed positive;

ii. true negatives (TN), which refer to correctly rejecting a negative condition;

iii. false positives (FP), also called false alarms and labeled Type I errors, refer to mispredicting a condition as being positive when its true condition is negative

iv. false negatives (FN), also called misses and labeled type II errors, which are cases where the
Figure 2.3: A visual representation of the confusion matrix and its four possible outcomes.

Based on these four possible outcomes one can express the definitions of recall and precision \[39\] as:

\[
Recall = \frac{tp}{tp + fn} \quad (2.2)
\]

\[
Precision = \frac{tp}{tp + fp} \quad (2.3)
\]

These two metrics often go hand-in-hand, and are in turn used to calculate the F-score \[39\]:

\[
F-score = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} \quad (2.4)
\]

In supervised learning classification tasks, accuracy is often the go-to metric for performance estimation. The formula takes into account all four of the above outcomes considered in the confusion matrix, and is expressed as:

\[
Accuracy = \frac{tp + tn}{tp + tn + fp + fn} \quad (2.5)
\]
These standard metrics are very useful in assessing model performance in machine learning. However, a model’s estimation capabilities can be good for a bad reason, or have poor performance due to know issues; below we take a deeper look into the problems of underfitting and overfitting, as these are common when using learning algorithms and should not be overlooked despite of a model’s performance.

### 2.2.3 Underfitting and Overfitting

Whenever we provide some data to a particular machine learning model, we are providing it with important details and features it should learn from, but we are also inevitably showing it noise. This is why the quality of the dataset is sometimes so impactful for the performance of the algorithm.

In statistics, fitting refers to approximating a target function. This terminology applies to machine learning as well, since in supervised learning, for example, we want an algorithm to approximate the unknown underlying target function for the output of the algorithm given its input - the training set.

If we use a particular algorithm and we can see that it doesn’t do a good job modeling the training data or generalizing to new data, then the algorithm is likely underfitting the data. However, underfitting is not discussed as often as overfitting as being a problem; this is because we’re more likely to have limited data to work with, making it a simpler (and in a sense, cheaper) solution to just try a different machine learning algorithm that performs better for the data we have.

On the other hand, the term overfitting is used when an algorithm models the training data too well. What this means is that the algorithm is likely learning both the important details and the noise in the training set, making it very successful for the particular concepts learned from the training data, but in turn doing a bad job generalizing for new, unseen data. Such a problem is bound to occur when one does what is called resubstitution validation: the model is trained with all the available data and is then tested on the same set of data as well; it is clear how this form of validation results in poor generalization capabilities when shown unseen data, which is why it is not often used.

Finding an appropriate fit is, in reality, trying to find a good trade-off between variance and bias. Imagine we would repeat the model building process many times - that is, to repeatedly train our model on different data each time. On one hand, the randomness that is inherent to these different training sets would result in the output models making a certain range of predictions. Bias is what measures how far off in general these models’ predictions are from the actual correct value; on the other hand, we call variance to how much these predictions vary across those realizations i.e. it refers to how much a model changes in response to the training data.
Scott Fortmann-Roe [40] depicts how bias and variance affect model predictions using a bulls-eye diagram, which you can find in Figure 2.4. The center of the target is a model that estimates the correct values with pinpoint accuracy. The farther away we get from the center of the bullseye, the less accurate the predictions are. Let's revisit the idea of repeatedly training our model with different data sets each time; we can represent these variations as separate hits on the target. Each blue dot on the target represents an individual realization of the model. In some of these realizations, our samples in the training data will provide good, accurate predictions, which will be closer to the bullseye. Conversely, in others, the samples in our training set may not constitute a good representation of what we want our model to learn, resulting in less ideal predictions, which could be due to a high number of outliers among those many samples, for instance. The scatter of hits in the target therefore represent the different realizations of our model.

We've seen before how we can tackle the problem of overfitting from the perspective of each of the algorithms. However, there are other methods which are very popular in addressing overfitting. Some of these methods can be performed regardless of the algorithm they are used in, since it involves pre-processing of the data.

One such way of avoiding overfitting is called hold-out validation. In this process, the available data is split into a train and test set in a way that neither of the two parts overlap (see Figure 2.5). This results in a more accurate estimation of the generalization performance of the algorithms that benefit from this form of validation. However, holding out part of the data is a luxury only affordable if one has
Figure 2.5: In hold-out validation, some of the data is held-out for validation to ensure the model is trained on unseen examples.

A big enough data set, which isn’t always true. Other problems with hold-out validation are that not all the available data is used for both training and testing, and that the results are highly dependent on the choice for the train/test split; since the instances included in the test set may be either too easy or too hard to classify, this can skew the results and lead to poor performance. This problem can be solved by repeating several hold-out iterations, but there’s no guarantee that some data wouldn’t be included in the test set several times; or on the other hand, that some data never became part of the training set, therefore leading to a poorer approximation between training set and the whole data set.
Related Work

Contents

3.1 Emotion regulation in ITS ........................................... 23
3.2 Emotion Regulation in Serious Health Games ......................... 27
3.3 Wizard-of-Oz (WoZ) Studies ......................................... 31
As we’ve seen, the problem of attempting to simulate human success in autonomous tutoring systems is something that goes a while back; several agents, models and architectures with strong psychological foundations have been developed to achieve this purpose. Thus, it is deemed important to understand how these systems regulate emotions, how they adapt and change over the course of a tutoring session, and how they decide whether it’s necessary or beneficial to intervene.

In the first section we describe and analyze several emotion-aware ITSs and what strategies they use to suppress or overcome negative emotions. Afterwards we look at serious emotion-regulation games and their health applications in hopes to replicate their success conditions and the usage of biological and physiological sensors to improve a person’s emotional coping strategies and mindfulness. Then we examine systems that resort to learning approaches in order to improve the effectiveness of ITSs methodologies in more complex scenarios.

### 3.1 Emotion regulation in ITS

#### 3.1.1 AutoTutor and Affective AutoTutor

AutoTutor is an intelligent tutoring system that teaches concepts of Newtonian physics to students using adaptive dialog in natural language to resemble a human tutor as much as possible. It models students’ knowledge levels by examining their textual responses and adapts its interaction towards them accordingly to provide a personalized experience. Its affective version features two implementations, the Supportive and Shakeup AutoTutors, which detect and act upon students’ affective and cognitive states. A set of affect-sensitive production rules map dynamic assessments of the students’ cognitive and affective states with appropriate tutor actions. Particularly, at any given turn of the dialogue, Affective AutoTutor keeps track of five major informational parameters that provide the foundations for affect sensitivity. Of particular interest are the two main strategies used by these alternate versions to regulate students’ negative affective states: the attribution theory and the cognitive disequilibrium theory, which address the feelings of boredom/frustration and confusion, respectively.

In a typical interaction with the student, AutoTutor replies to his/her response textually as well as with an adequate facial expression. Afterwards, in case the student is neutral, the agent’s reply is a non emotional remark. Otherwise, if it senses the student manifested a negative emotional state, it delivers an emotional statement. If this negative emotional state is perceived as boredom or even frustration, the agent acts according to the attribution theory.

On one hand, the Supportive AutoTutor blames the material or other external factors as the cause of the student feeling boredom or frustration. For example, the agent might reply with “Maybe this topic is getting old. I’ll help you finish so we can try something new”, in an attempt to indirectly address student’s
boredom while also focusing on a new topic, hence avoiding the student becomes disengaged from the learning experience. On the other hand, the Shakeup AutoTutor points to the student itself, in an attempt to make him/her aware of their own emotional state. Both these approaches aim at improving the students’ awareness of their emotions.

In line with the cognitive disequilibrium theory, an attempt is made to address the emotional state of confusion in both Affective AutoTutor versions. To state another example, the Shakeup AutoTutor might say “You are not as confused as you might think. I’m actually kind of impressed. Keep it up”. This strategy makes students aware of their confusion, as well as helping them break from it by, again, shifting focus to something else, like their good track so far in this example, and motivating them to move forward.

3.1.2 EER-Tutor

In 2008, given the affective gap in Intelligent Tutoring Systems, Zakharov et al. [41] developed an affect-aware pedagogical agent persona for an ITS that teaches database design skills, EER-Tutor [42]. The project uses feature tracking technology to learn the user’s affective state in a dimensional approach (which we’ll explain further) that is then incorporated into the ITS, called EER-Tutor. The following subsections explain the decisions made by the researchers as well as the implemented features and strategies used, ending with an overview of the experiment and its results, in what is a significant take on ITS and the connection between the cognitive and affective processes of humans.

3.1.2.A Identifying Users’ Affective States

When deciding how to model the affective states in their research, the authors had to choose between two major theoretical approaches to the study of emotion. In a categorical approach to emotion, an estimate of three to twenty basic emotions are considered that are combined to produce all emotional states that people experience. On the other hand, a dimensional approach models emotional space as having two (perhaps three) underlying dimensions where the whole range of human emotions can be arranged. The most commonly considered dimensions are valence (ranging from happy to sad) and arousal (ranging from calm to excited). The authors decided to adopt the dimensional approach because, for one, this approach requires no classification of the emotional states as belonging to specific categories, which eliminates difficulties in emotion modeling. The continuous nature of the valence dimension in this approach also facilitates the author’s choice of implementation of their feature tracking algorithm.

In order to assess a person’s affective state given the dimensions of the aforementioned approach, the authors based their facial feature tracking techniques on action units in the Facial Action Coding System.
System (FACS) [43]. For example, positive affective valence is indexed through a decrease in distance between the corner of the mouth on one side of the face (corresponding to action unit #4 in FACS), an action that results in a smile. The authors also developed a feature tracking algorithm that includes five steps: face region extraction, iris detection, outer eye corners detection, mouth corners detection and inner brow corners detection. This information is then used to make decisions on the basis of observed changes throughout the session rather than attempting to determine the affective state in every frame; every time an update is received from the feature tracking code, the affective state is updated to register transitions between negative, neutral and positive affective states - again, made possible by the continuous nature of the valence axis in the dimensional approach.

3.1.2.B Affective Pedagogical Agent for EER-Tutor

In respect to users’ preferences, the author’s decided to use two male and two female characters (avatars) designed to appear between 20 and 30 years of age. The characters used text-to-speech engines to generate verbal narrations along with realistic lip-sync movements and respective male and female voices. The agent's persona is governed by a set of rules that encode logic of session history appraisal, and they assume two things: that the continuous lack of cognitive progress is tied to a negative affective state; and conversely, a satisfactory progress is tied to a positive affective state.

One thing the authors decided to do was to couple each rule, which has a set of corresponding feedback messages, to a numeric value which triggers a change in the agent’s affective appearance. So if, for example, the user reaches the correct solution of an exercise or task, the agent responds with a congratulatory message and a cheerful smile. However, in the absence of these affect-triggering changes, the agent’s affective state always gravitates towards the neutral state, similar to what happens with human emotions.

Another interesting feature of the agent is that while it is capable of distinguishing between positive and negative affective states, it only addresses steady negative affective states. The reason for this, based on other theories, is that the state of positive flow may be disrupted by making the subject aware of the flow - which is why the agent does not need to interfere unless there is a negative affect. This interference could potentially break the mood or distract the user from his/her task. On the other hand, making subjects aware of their negative affect could be beneficial, as it may distract them from their negative feelings and help them move towards their objectives. But what if making users aware of their negative feelings worsen the situation because they find the interventions irritating? With this in mind, the authors designed the agent to only provide affect-oriented content if the subject’s facial feature tracking data indicates the dominance of the negative affective state.

Some examples of feedback messages that the agent uses for affect-oriented feedback, intended to
address the user’s negative feelings and to be empathetic towards the user in adequate contexts are as follows:

- “I’m sorry if you are feeling frustrated - it’s just that some of the problems demand a lot of work.”
- “I apologize if you feel negative about this practice session - some of the solutions are quite complex.”

3.1.2.C Experiment and Results

In their experiment, they asked users to give their feedback in a free-form questionnaire. There were positive and negative impressions, but the authors concluded that in general, approval of the pedagogical agent’s presence in EER-Tutor dominated the questionnaire responses. They also added that while the agent’s uptake wasn’t unanimous, “the evaluation results advocate the presence of affective pedagogical agents, with the affect-aware agent demonstrating superiority over its non-affective counterpart”.

3.1.3 A web-based learning system

Strain and D’Mello [10] developed a web-based learning system that analyses the effect of cognitive reappraisal on students’ self-reported emotions and performance results. Two variables are considered for the evaluation: valence and arousal. The authors used different cognitive reappraisal conditions to test their system.

Participants were randomly assigned into three different groups, each with different reappraisal conditions: deep reappraisal, shallow reappraisal, and no reappraisal (control). The participants in the first
two groups were asked to imagine they were applying to work at a very powerful law firm and had to perform a special task to get the job; those in the deep reappraisal group had to read a document and evaluate its comprehensibility and those in the shallow reappraisal group had to check a document for typos and grammatical errors. The participants in the control group had no instructions regarding cognitive reappraisal.

Then, in a web-based learning session, participants were asked to learn about the U.S. Constitution and Bill of Rights, answer questions about what they learned, and report their affective states based on valence and arousal at the end of each page of these documents.

The results (see Figure 3.1) favored the usage of cognitive reappraisal. Participants in the deep and shallow groups reported higher arousal that those in the control group. The same happened for valence states such as alertness and engagement. Overall, it was concluded that cognitive reappraisal enhances comprehension abilities and proved to be a useful method for emotion coping and regulation in tutoring systems.

3.2 Emotion Regulation in Serious Health Games

The usage of virtual environments - such as video games and the internet - as more cost-effective treatment approaches in health and medical applications has become increasingly popular in recent years [7, 8]. Serious games are a particular example of those environments, and in this section we take a look at several games which perform emotion regulation strategies to help players manage their affective state.

3.2.1 EmoRegulators

EmoRegulators [14] is a serious game that helps adolescents learn to perform emotion self-regulation through a series of sessions based on a technology-enhanced version of the BEAR protocol. The original BEAR [15] was developed by the Israel Center for the Treatment of Psychotrauma (ICPT) as a group intervention for building resilience in traumatized children, and features a series of sessions which aim to strengthen psychological, cognitive and social regulation abilities among those children. As such, it makes sense that the protocol had to be made suitable for individuals rather than groups, and for adolescents rather than children, as well as being converted to be used in a technological application.

In order to assess the user’s emotional state, the game uses bio-sensors from PLUX. There are four different sensors: an electrocardiogram (ECG) sensor that measures heart rate (HR), one for electrodermal activity (EDA), and two electromyography (EMG) sensors that measure muscle strength to be placed on the biceps and trapeze. These sensors are attached to the user’s body and to a hub which
Figure 3.2: EmoRegulators system components.

sends the signals via Bluetooth to the OpenSignals application, which makes the bridge between the sensors and the game application, as Figure 3.2 shows.

Since there is a lot of information being sent by the sensors, the physiological data is currently processed in order to send only a summary of the relevant physiological information to the Emoregulators application in Unity. For example, the raw heart rate signal is converted to a measure of beats per minute (BPM), which is then sent to Unity. All this information is stored and logged by a component of the application called the sensor manager. This is relevant because the correct or incorrect practice of the exercise is determined by this stored information. For example, a user successfully completes the Active Shaking Meditation exercise if his measured BPM during the dancing exercise was at least 50% higher than his baseline (state of no activation) BPM.

The single player simulation currently consists of two sessions:

Session 1 (Introduction): Presents the topic of coping resources identification via experimental, playful and fun exercises. It aims at increasing the awareness of self within the game.
Session 2 (Physical Regulation): Focus on experimental exercises in order to practice relaxation coping skills and provide awareness of one's bodily sensations. During game play, users will perform breathing, active/shaking and Progressive Muscle Relaxation (PMR) exercises and reflect about their safe places and their internal sensations.

During these sessions, the player earns points by completing the exercises and can see them at any time. The points are displayed below their personal box, the I-Box, which they learn about and customize in the introduction session. Players can also see their heart rate, which makes them aware of their physical state.

One of the exercises in the first physical regulation session refers to facial mindfulness. In this exercise (see Figure 3.3 above), the player is told to focus on the many parts of his face, such as the forehead, chin, mouth and eyes. The player is then asked to reflect on whether each of those parts are either tense or relaxed, as well as whether they have any other feelings. Afterwards, the player is asked to notice his facial expression without changing it. Finally, the player is asked to color his avatar’s face red where he previously felt tense or blue where he felt relaxed. This is an example of a calm, low-activity exercise which has the player reflect about his internal sensations.

In contrast with the former example, another exercise asks the user to dance for about 3 to 5 minutes while trying to use all his/her body parts. The player’s avatar body is shown to suggest to follow its movements, but the player is told he can dance freely if he’d rather do so. The player is also reminded this exercise is very rewarding in terms of points - which are attributed to the user whenever he/she successfully completes an exercise. This is the Active Shaking Meditation exercise mentioned earlier and it is an example of a high-activity exercise that makes users loosen up and feel their whole body. These exercises have different purposes but they all share the common goal of performing physical regulation.
3.2.2 PlayMancer

PlayMancer [44] is an EU initiative to develop a video game prototype for treating specific mental disorders, namely eating disorders and impulse control disorders. It is based in an interactive scenario called Islands, where the ultimate goal is to increase emotional self-regulation skills in patients and also to increase their self-control towards impulsive behaviors. The game is split into three mini-games. In all of them, it’s the player’s emotions that dictate the difficulty of the mini-game.

1. The face of Cronos - a game where the player has to climb a cliff while administering his/her resources and avoiding obstacles, which are produced according to the player’s emotions.
2. Treasure of the Sea - a game where the player swims under water, gathering artifacts and balloon fish while also having to manage his/her oxygen level to stay alive and keep playing. Also, the difficulty of diving depends on the player’s emotions.
3. Sign of the Magupta - a relaxation mini-game in which a constellation of stars is drawn according to the level of calmness of the player.

<table>
<thead>
<tr>
<th>Game task: User requirements</th>
<th>Therapy goals:</th>
</tr>
</thead>
<tbody>
<tr>
<td>The face of Cronos (climbing)</td>
<td>Lack of stress management</td>
</tr>
<tr>
<td></td>
<td>Low tolerance to cope with adversities</td>
</tr>
<tr>
<td></td>
<td>Impulsive behaviours</td>
</tr>
<tr>
<td></td>
<td>Strong negative emotional expression in front of minimal stimuli</td>
</tr>
<tr>
<td></td>
<td>High physiological reactivity in front of stress</td>
</tr>
<tr>
<td></td>
<td>Lack of boredom management</td>
</tr>
<tr>
<td>Treasures of the sea (diving)</td>
<td>Lack of stress management</td>
</tr>
<tr>
<td></td>
<td>Low tolerance to cope with adversities</td>
</tr>
<tr>
<td></td>
<td>Impulsive behaviours</td>
</tr>
<tr>
<td></td>
<td>Strong negative emotional expression in front of minimal stimuli</td>
</tr>
<tr>
<td></td>
<td>High physiological reactivity in front of stress</td>
</tr>
<tr>
<td></td>
<td>Lack of boredom management</td>
</tr>
<tr>
<td>Sign of the Magupta (relaxation)</td>
<td>Strong negative emotional expression in front of minimal stimuli</td>
</tr>
<tr>
<td></td>
<td>High physiological reactivity in front of stress</td>
</tr>
<tr>
<td></td>
<td>Lack of stress management</td>
</tr>
</tbody>
</table>

Figure 3.4: Game tasks, user requirements and therapy goals in PlayMancer.
This information regarding the person's emotions is gathered through bio-sensors (galvanic skin response, heart rate, heart rate variation) and facial gestures, and speech based emotion recognition is used to decide which skills and attitudes should be changed in order to achieve each task's therapy goals (see table in Figure 3.4).

3.3 Wizard-of-Oz (WoZ) Studies

The usage of learning architectures in intelligent agents has become increasingly popular over the years. These agents are capable of performing very well in more complex situations, given they've been trained properly. The catch, though, is that these systems usually perform very poorly in the beginning, getting better as they have more and more interactions.

In this section we take a look at several studies that use the WoZ paradigm.

3.3.1 Learning from the Wizard (LfW)

Bradley et al. were supposedly the first to propose and develop an algorithmic and experimental framework for human-robot interaction through learning from demonstration (LfD) specifically for learning social behaviors [45]. Particularly, they did the demonstrations within a Wizard-of-Oz (WoZ) paradigm, hence referred to as Learning from the Wizard (LfW).

There are three basic stages involved when conducting an experiment via LfW. First are the Wizard of Oz demonstrations, then a learning algorithm is developed and applied based on the data collected in the demonstrations and, finally, evaluations of the learned policy are held and the algorithm may be polished according to the success of the agent's standalone performance.

Usually in LfD, the person controlling the agent in the demonstration phase is not present in the last phase. Thus, to avoid mismatches between the training (demonstrative) and testing (evaluative) environments, participants should not be aware of the demonstrator's presence in the first phase. WoZ demonstrations hold true to this in order to facilitate the future mapping of behavior between the demonstrator controlling the agent and it acting autonomously.

Wizard of Oz refers to a scenario in which an agent is secretly controlled by a human puppeteer as it interacts with a human participant. In the demonstration phase, Bradley et al. propose that the demonstrator's observations of the interaction should reflect the sensory input that the agent's learning algorithm and resultant behavioral policy will have available, as long as this input is understandable to the demonstrator.

Once enough demonstrations have been performed, a learning algorithm is applied to create a map-
mapping from the space of possible observation histories and the space of possible actions to a probability. This mapping is broken down into two steps: mapping from observation history to features and mapping from features and an action to a probability. The first step is particularly complex in the context of learning social interaction, and here's why.

The true state space of a person in social interaction is unknown and intractably complex because it involves the full neurological state that gives rise to their behavior. Social interaction involves memory and inference about information never observed. A possible way to address this problem is to map from the agent's observation history to a relatively simple state description, which might be sufficient to provide enough context to emulate the demonstrator's decisions with sufficient fidelity to produce the desired interaction effects, such as engagement, joy, learning and others. The most common approach for deriving such a state signal is to hand-design features that are thought to provide context for the demonstrator's action.

Finally, when a satisfactory policy has been derived from demonstration data, the policy can be formally evaluated. The authors describe two categories of analysis which they explore in two ongoing studies, which we briefly describe next. The first type directly compares the learned behavior to the demonstrations provided. This provides a great idea of how well the demonstration behavior was transposed to LfD. The second type compares the learned behavior to other types of interaction, such as human-human interaction or one where the agent's behavior was fully computed from other techniques, to name a couple.

As mentioned in the above paragraph, the authors were conducting two ongoing studies at the time of writing using each of the above described evaluation techniques. The first study is a nonverbal, embodied and social Turing Test, where the authors intend to find out whether an LfW approach for social interaction makes the agent's interactions more believable, to the point of being comparable to those of a real human person.

As for the second study, the general idea is to have participants interact with the agent in the context of mutual play with an Android tablet app. During the interaction, the child's actions upon the app will be used as an additional source of context for agent behavior. Then, the child-robot-app interaction with the learned policy will be compared against the child interacting with the app alone.

While these were ongoing studies, it is concluded by the authors that a WoZ control avoids an early-learning period of low-quality behavior of the agent, since demonstrations capture teacher behavior better than asking him/her to describe it, and since non-roboticist humans can easily provide demonstrations to the agent, which supports the usage of the Wizard-of-Oz paradigm in the context of Intelligent Tutoring Systems and social interaction in virtual environments such as that of serious games like the ones described in previous sections.
3.3.2 Restricted-Perception WoZ Studies

Sequeira et al. [46] propose a methodology to create social interaction strategies for Human-Robot Interaction (HRI) based on restricted-perception Wizard Of Oz (WoZ) studies. The interesting core idea is restricting the wizard’s perceptions over the environment and the behaviors it controls according to the agent’s inherent perceptual and acting limitations. This methodology is divided into three phases, which we detail next.

3.3.2.A Data Collection

The first phase is data collection. The main purpose here is to prepare and perform restricted-perception WoZ studies by gathering useful knowledge about appropriate interaction strategies to be considered. The idea is to let humans that are experts on the given task to perform several interaction sessions with prospective end-users of the system. The collected data is used to build a set of task-related artificial intelligence modules, referred to as the Task AI.

It starts with the use of mock-up studies. These prepare the WoZ studies and inspire the development and implementation of the different system components controlling the agent’s interaction strategy. After acquiring expert knowledge from the mock-up studies, the agent’s actions and perceptions in the task are devised. Due to the complexity of processing all the agent’s input data, many of the interaction aspects regarding the task itself are simplified in terms of perception and behavior. For example, if the task requires the agent to point to someone or say something specific, one can create macro operators that encode the necessary low-level behaviors. There can also be considered specialized planning and decision making algorithms that address specific problems, such as playing a game. All of these mechanisms compose the Task AI mentioned before, and this approach simplifies the expert’s perception and the decisions he has available.

Once the Task AI has been implemented, appropriate interaction strategies can be discovered for the agent by performing the WoZ studies. This is the interesting, fundamental part of this methodology in question, as it is what allows the generation of the interaction data that can later be used to encode behavior rules and apply machine learning techniques to automatically extract appropriate interaction strategies.

Given how it’s not practical to design behavior for every predictable situation, the standard WoZ technique is significantly important to help achieve that. However, it has many complications when trying to extract useful behaviors from the expert’s interactions. Many perceptual and acting limitations of the agent are usually disregarded by letting the wizard completely observe the interaction, making it hard for the agent to correctly interpret the environment and act autonomously.
An example would be that the lack of a very accurate speech recognition and interpretation module could cause the agent to not understand and interpret what would be said in an interaction, therefore responding poorly and out of context. This is where the restricted-perception addresses these issues, by limiting what the wizard extracts from the task’s environment. In that respect, the Task AI enables ML algorithms to discover complex behavioral patterns exhibited by the experts during the interaction. The experts also undergo a training phase to get used to the agent’s capabilities and their feedback is used to iteratively refine the user interface prior to the studies.

3.3.2.B Strategy Extraction

In this next phase, the idea is to build an interaction strategy controller for an agent based on the previously collected data, corresponding to the Strategy Extraction phase, where they try to “infer” the decision process used during the restricted-perception WoZ studies.

One of the underlying problems in WoZ studies is the correspondence problem, where a direct mapping between sensors and actions of the agent and those of a human is not possible. The restricted-perception approach mitigates this problem, as previously explained, helping to significantly reduce the complexity of finding a correspondence between the task’s state as observed by the wizard and the information available to the agent. Because this technique only helps to mitigate the problem, a hybrid interaction strategy controller is proposed.

The idea is that a data-driven ML-based module and an event-driven Rule-based module compete for the guidance of the agent’s interaction behaviors. The rule-based module is responsible for modeling well-known strategies in the form of behavior rules, such as If-perceptual state-Then-interaction behavior rules, that are automatically activated at specific times during the interaction. Namely, certain rules are manually designed based on consistent practices employed by the human experts during the interactions. They also encode domain knowledge rules with information gathered by the Task AI, like triggering some behavior when some task milestone is achieved, for example.

The ML-based module is responsible for automatically discovering complex situations that may have arisen during the interaction sessions and for which it is hard to explicitly create behavior rules. In this methodology, interaction strategies are learnt through a mapping function between the agent’s state features and interaction behaviors given the wizard demonstrations in the restricted-perception WoZ studies. Similar to how the wizards have the responsibility of choosing which behavior to trigger and when to trigger it, so does the ML-based module. For this matter, ML algorithms are chosen to learn the mapping function, by using classification or clustering algorithms, for example.

A depiction of this module can be seen in Figure 3.5. First, there’s a Data Preparation phase involving the transformation of the collected demonstrations into a data-set of state features-behavior pairs,
Figure 3.5: A depiction of the ML-based module processing

referred to as training instances. Then, the Training phase learns a mapping function encoding the observed interaction strategies from the given data-set. And finally, the module may choose an appropriate interaction behavior at run-time when requested, given the agent's perceptual state.

3.3.2.C Strategy Refinement

In the last phase of the methodology, evaluation studies are conducted to assess the performance of the agent being autonomously controlled while interacting with others during the given task. Two mechanisms are used: active learning and corrective feedback.

Regarding active learning within ML, this technique attributes the learner the responsibility of querying an expert about specific inputs for which the agent needs to learn an output or improve his accuracy. Methods can be created to automatically identify areas of the agent's perceptual state that weren't as explored during the WoZ studies. Then these data points can be used to ask experts what interaction behaviors should be performed in that specific context. Also, mechanisms can be built within the ML-based module that identify data inconsistencies.

Corrective feedback may also be provided by a human expert during the evaluation study, again compliant with the restricted-perception conditional. The idea is for the expert to observe the output of the strategy controller while the agent is interacting with the human subjects, allowing him to accept the selected behavior or suggest another one, thus refining strategies that may have been incorrectly encoded or learned during the Strategy Extraction phase.

3.3.2.D Practical Use and Results

The authors implemented this methodology in the context of EMOTE - a project that aims to develop novel artificial embodied tutors capable of engaging in empathetic interactions with students in a shared
physical space - and MCEC - a multiplayer, collaborative game based on the serious game EnerCities that promotes strategies for building sustainable cities - and concluded that the generated interaction strategies allow the students to be engaged in the social interaction with the agent and perceive it positively in regard to its empathetic capabilities. It is safe to say that such a methodology is a great improvement to what is likely the hardest challenge in applying a WoZ paradigm - mapping between the expert’s observations and the agent’s perceptive capabilities - and it should definitely be considered when attempting to build successful interaction strategies applied to learning in ITS.
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 Emotion Regulation: Defining Appropriate Strategies for EmoRegulators</td>
<td>39</td>
</tr>
<tr>
<td>4.2 Activation of the Emotion Regulation Strategies</td>
<td>41</td>
</tr>
</tbody>
</table>
In the section dedicated to related work, we had a look at the three types of work that will be intertwined to form the proposed solution. For us to solve the problem of how to create an intelligent, adaptive system in EmoRegulators, capable of being aware of, understanding and regulating players’ emotions during an interaction session, we need to solve two distinct subproblems. First, we need to decide what may be some appropriate emotion regulation strategies applicable in the context of EmoRegulators based on literature introduced in chapter 2. And second, we needed to solve the problem of how these strategies should be activated in EmoRegulators. It is hoped that by using restricted perception Wizard-of-Oz studies, introduced in chapter 3, one can help emotion-aware intelligent tutoring systems to better replicate the behavior of expert human tutors. In the solution, the necessary steps to solve these problems are taken in order to make the in-game facilitator into an intelligent tutoring system rather than having a scripted behavior.

4.1 Emotion Regulation: Defining Appropriate Strategies for EmoRegulators

The task of deciding which emotion regulation strategies could be suitable in the context of EmoRegulators required that some research into the game components and exercises was conducted first. Early thoughts on how to solve this subproblem were that it could be interesting to experiment with both exercise or activity related strategies, as well as ones which could be used at any given moment in the game. But these strategies are supposed to complement the standard behavior of the facilitator, meaning each of them individually would likely occur few times during the full length of a game session. This means exercise-specific ones would see so little usage that it would be hard to assess their impact on gameplay.

As such, focus was shifted towards defining strategies that could be activated in any moment of the session while remaining relevant. Literature on many views on emotion regulation was studied and it was found that the attribution theory and the cognitive reappraisal theory, presented in chapter 2, could help solve the problem of keeping players engaged during a game session of EmoRegulators. The decision to experiment with these strategies was inspired by their usage in other successful Intelligent Tutoring Systems such as AutoTutor and Strain and D’Mello’s web-based learning system [10] presented in the previous chapter.

In its classical version, the facilitator in EmoRegulators interacts with the player verbally, with a calm voice, following a script that explains each exercise. The least intrusive way to intervene when the sensors readings say the agent should do so is to have a set of empathetic or motivational interactions that it can choose from and say at the appropriate moment.

In the current version, strategies meant to apply the attribution theory will attempt to blame the
feelings of frustration or boredom on external factors, such as the exercise or the agent itself, similar to what AutoTutor does. For example, if the sensors readings indicate the player might feel bored, the agent can say something like “This exercise is a bit long...” or “Am I talking too much?”. Both examples, depicted in Figure 4.1, attribute the cause of boredom to other things, alleviating that pressure from the player, while making him/her aware of their emotional state. The first example can be used to keep the player engaged in a lengthy exercise, while the second one can recapture the player's attention during a tiresome explanation.

In cognitive reappraisal, the idea is to say something that causes the player to acknowledge their negative emotional state and then re-evaluate the situation to overcome it. This is a very beneficial approach for two different reasons. The first is that it can potentially help the agent regulate the player’s emotions, which is one of the goals for the agent. The second is that, as an added bonus, it makes the player aware of his/her emotions upfront which, remember, is important in a game like EmoRegulators that is meant to teach players to be aware of, and control, their own emotions.

In one of the strategies that follow this theory, the facilitator says “Even if you’re not feeling very motivated, remember: you can earn a lot of points if you make this little effort!”. This does two things: for one, it helps the user recognize their negative response to the exercise by suggesting he may be lacking motivation to continue; and, secondly, it emphasizes there is a reward for completing the exercise in an attempt to refocus the player’s motivation to pursue additional points and to push through the session with a fresh mindset.

As an example of how this interaction could prove useful, consider an exercise where high physical intensity is required, such as the one where the player is required to dance for a few minutes. Since
players can opt to simply ignore such an exercise by not moving or altering their physical activity, if the expert has information about the points available, and the sensors’ readings say the HR is low, he/she can chose this interaction to try to convince the player to do the exercise as intended (by actually moving and dancing) increasing their HR and potentially improving the player’s emotional state. This is a double-edged sword, however; if the sensors indicate more than just a low HR - boredom, for example -, it might be better to take a different approach and let the player ignore it, because they might not like the exercise they’re about to do and maybe ignoring it will suppress or eliminate this feeling. This is entirely the expert’s call, but making this interaction available can give the expert leverage for persuading the player to go through with the exercise - if that is the correct decision.

In addition to the attribution theory and cognitive reappraisal strategies, we decided to incorporate additional generic empathetic remarks that celebrate the player’s performance in the game - “Good job!” and “Proud of you!” are examples of that. It is hoped that this can help maintain a positive arousal and attitude during the game, reducing the likelihood of feelings such as boredom and frustration taking place. The remaining interactions “Almost there!” and “You Can Do It!” are a motivational version of the aforementioned ones, implemented with the same purpose in mind; finally, “You’ll Feel Better” also makes use of the cognitive reappraisal by highlighting that the purpose of EmoRegulators is to make players feel better, and that in doing these exercises they are working towards that goal.

The initial interactions were reviewed by the domain consultant who fulfilled the role of the expert in preliminary demonstrations (this is explained in more detail in the section regarding restricted-perception WoZ demonstrations) and by a psychologist. Their feedback was that some of these emotion regulation strategies were too long and that it could potentially make them ineffective. In those strategies where it was possible, they were reformulated to become shorter and straight to the point while maintaining the essence of their respective theories (attribution theory and cognitive reappraisal) where these applied - the more generic ones didn’t require this consideration. As an example, one strategy was to say “I’m sorry... Am I talking too much? Maybe you’ve understood this exercise already. Should we start?”; indeed, this could be reduced to only the “Am I talking too much?” segment, which is the part that effectively applies attribution theory. This new iteration of shorter strategies were revised and approved by the domain experts and are the ones being used in this system at the time of writing.

4.2 Activation of the Emotion Regulation Strategies

Auto-tutor, a landmark in Intelligent Tutoring Systems, has the Support and Shake-up versions, which are affect-aware. In these systems, the activation of the emotion regulation strategies that the agent uses are decided according to a set of rules which guides this decision-making.

Looking at the literature, it was decided it could be interesting to combine the lessons taught by these
systems of what strategies can be used for emotion regulation, with the idea that the activation of these strategies should be entirely guided by human decision-making - specifically that of an expert - instead. This led us to explore Wizard-of-Oz studies but, more specifically, restricted-perception Wizard-of-Oz studies.

4.2.1 Methodology

As explained by the authors of the work [46] that explores this approach, by restricting the perceptions of the human acting as the expert in the demonstrations phase to match those of the agent, the learning curve is steeper (in the technical sense of the expression, meaning more is learned quicker). Our methodology was developed to use a restricted-perception WoZ technique to learn how to apply emotion regulation strategies in EmoRegulators.

In the case of EmoRegulators, the agent benefiting from the restricted-perception learning is the conversational instructor, also called facilitator, that is responsible for explaining the exercises and tasks in the game. The goal is to teach the agent to have tutor-like decision making in a tutoring scenario, instead of following a script, so that it incorporates some level of interactivity and dynamism into its dialogue system. This is done by allowing the wizard to effectively control the agent, typically choosing from a set of possible strategies during an early demonstration phase.

We first needed to identify the information available both in the game and provided by the physiological sensors, and determine which features would or not be relevant to help the expert (and the system), to make decisions on when to perform the emotion regulation strategies, as well as if any additional ones besides these would be necessary. We decided in favor of adding a few others besides the ones given by sensors and game state, which are calculated from existing ones - but a better explanation of these features is provided in the following chapter.

A fundamental requirement and necessary assurance in conducting restricted-perception WoZ studies is to enforce that the human expert has access to the same information that the system will have to make decisions. For this reason, a separate WoZ interface was carefully designed, implemented and interconnected to the EmoRegulators game to ensure this requirement was respected. The reader can refer to section 4.2.3 to see how this interface was designed exactly. This interface was then used by the expert while conducting the restricted-perception WoZ demonstrations depicted in the leftmost parcel of Figure 4.2, the phase when data collection takes place.

After this demonstrations phase, the gathered data is pre-processed before being used to train an automatic interaction model using machine learning techniques. Finally, this model can optionally be put to test against new players without the presence of the wizard and using just the learned interaction model instead, which would correspond to step (c) of Figure 4.2. Steps (a) and (b) of this restricted-perception Wizard-of-Oz paradigm were implemented in this work with the previously discussed set of
strategies meant to keep players engaged in EmoRegulators.

4.2.2 Game State and Affective State Representation

One very core aspect of this work was to define which features or information would be used to model the state of a session of EmoRegulators. These features would play a leading role both in the restricted-perception WoZ demonstrations and in the training phase. Furthermore, these features would need to be representative of the game state, as well as the affective state, in order to build a solid set of features for the successful application of emotion regulation strategies in EmoRegulators.

In regards to game state, the session, exercise, and points gained by the player were collected during play time. This information would help guide the expert’s decision-making during demonstrations by providing context for the data collected from the game at every instant. This is complemented, of course, by features corresponding to the data collected from the bio-sensors. This information was also later complemented by collecting, for a given instant, time elapsed since the beginning of a game session and since the beginning of the exercise being done as well.

Finally, a few additional features calculated from these first ones went into modeling game state. One of these is a feature that represents arousal using some of the physiological data. Despite there not existing a direct mapping between emotions that suggest lack of engagement, such as boredom and frustration, and the data that can be collected from the bio-sensors in EmoRegulators, thankfully it has been shown that a significant correlation exists between psycho-physiological arousal (i.e., HR and EDA) and self-reported gameplay experience, which includes the feelings of frustration, boredom, and others [47]. Other claims have been made about the correlation of higher EDA readings and a higher level of frustration [47, 48], and other studies report in favor of a correlation between higher HR and higher player arousal [48, 49].
Therefore, arousal, meant to help the specialist understand the player’s emotional state, was implemented as being represented on a 0 to 10 scale. This was done so that an arousal meter using this scale could be shown in the restricted-perception WoZ interface. As such, the following steps were taken to calculate this value:

i. Aggregate data from all past sessions of EmoRegulators (specifically, those conducted with players in past work on the game) for both HR and EDA;

ii. Calculate the mean and standard deviation for those variables;

iii. With every data update, in real-time, calculate the Z-Score \[ z_{HR} = \frac{x - \mu_{HR}}{\sigma_{HR}} \] \[ (4.1) \] for heart rate where \( x \) is the current measurement of HR, and likewise for EDA.

iv. Adjust to a 0 to 1 value range and then scale this value to a 0 to 10 range.

However, we had several issues gathering EDA data during development. An additional step five, which would weigh HR and EDA based on how they affect arousal could not be performed since we were not able to collect reliable EDA sensor data. This means EDA was removed from the calculation of arousal in the meter, which should remain plausible given that it has been proved that HR correlates to arousal \([48, 49]\).

Our initial idea was to simply use the above calculation and normalization of Z-score to determine a scaled value for arousal between 0 and 10 initially. However, after testing this feature we noticed that many of the heart rate values occurred moderately above the mean of HR values obtained from previous EmoRegulators tests with teenagers; this resulted in a scale where in reality all HR values would fall in the 4 or 5 to 10 range according to our adjustment of Z-score, meaning too many high values were being represented as a 10 in the scale simply because they didn’t occur often in those original tests the calculation is based on. However, this calculation was still a good metric for arousal for EmoRegulators specifically.

For this reason we decided to additionally make adjustments to which values represented both ends of the arousal scale. First, we investigated whether there was research done on what are common values for heart rate values per age. Surely enough, we found “Normal ranges of heart rate and respiratory rate in children from birth to 18 years: a systematic review of observational studies”, a systematic review by Fleming, Susannah, et al. \([51]\) which includes data from 69 studies and heart rate data for over 140,000 children ranging, as the title makes evident, from birth to 18 years old. The web appendix of this study
has tables with the values, where one can find that people aged 13 to 18 years old have a normal heart rate ranging from 60 to 100 beats per minute. Since EmoRegulators is meant for adolescents in an age interval (14 to 18) that is a subset of this range, we used 60 as the base value for the arousal scale.

As for maximum heart rate, while not perfectly accurate, the majority of sources still cite Fox III, Samuel M. and John P. Naughton’s “Physical activity and the prevention of coronary heart disease.” and the suggested formula [52] for maximum heart rate calculation there included. The formula can be expressed as a function of age, as follows:

\[
MaximumHeartRate(age) = 220 - age
\]  

(4.2)

where \( Age \) is expressed in years. So according to this formula, we took the youngest possible age for adolescents in our study, which was 14, and used that value to calculate maximum heart rate, as the lowest age is the one that maximizes the function. As such, we defined the ceiling of our arousal meter as 206. Finally, we simply subtracted the maximum value for heart rate from the minimum one we chose and divided by 10 to calculate the step:

\[
Step = \frac{MaximumHeartRate(14) - 60}{10}
\]  

(4.3)

\[
Step = \frac{206 - 60}{10}
\]  

(4.4)

So we used the resulting value of \( step = 14.6 \) to map heart rate values to the values in the arousal scale ranging from 0 to 10. Finally, we weighed the Z-score calculation at 50% and this calculation at 50% as well (rounding up), so that both affect a calculation for arousal equally. To clarify, the step is what we call every interval between consecutive values in the arousal scale. For example, if at a given moment heart rate is measured at 98, the arousal value for this calculation would be given by \((98 - 60)/14.6 \approx 2.6\); we would then weigh this with the resulting arousal given by Z-score and round up to get the final value. This final value of arousal was the one used as a feature in the work.

\[
PercentVariance_{HR} = \frac{HR_t - HR_{t-1}}{HR_{t-1}}
\]  

(4.5)

Besides the raw data that could be gathered from EmoRegulators, there were other features besides
time and arousal which were considered to be fed to the algorithms in the training phase to come. Additionally it was decided that, at a given instant when data was collected, there should be a calculation of the average of the 10 most recent samples leading to that moment for heart rate, EDA, bicep activity and trapeze activity as features. So this resulted in four additional features called averageHR, averageEDA, averageM1 (for bicep activity) and averageM2 (for trapeze activity). These features could potentially come to play in the training phase for specific approaches which we present later in this document; since not all machine learning algorithms take into account feature dependency or connections between different features, this is one way to include variables that account for some sort of connection between them. Finally, the percent variance for HR at instant $t$ between each two consecutive heart rate readings was calculated as a feature for each data observation, according to the formula in 4.5. An example containing almost all of these features (because some differed from one training approach to the other) is shown in table 4.1 below.

Table 4.1: A table containing the features used in two of the training approaches. M1 denotes the trapeze, M2 denotes the Bicep, and HR%Var refers to HR Percent Variance. The indexes associated with each row refer to the number of the sample in that window of 10 samples.

<table>
<thead>
<tr>
<th>Session</th>
<th>Exercise</th>
<th>Time</th>
<th>Points</th>
<th>HR</th>
<th>EDA</th>
<th>M1</th>
<th>M2</th>
<th>HR%Var</th>
<th>Arousal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session1</td>
<td>Exercise1</td>
<td>Time1</td>
<td>Points1</td>
<td>HR1</td>
<td>EDA1</td>
<td>M11</td>
<td>M21</td>
<td>HR%Var1</td>
<td>Arousal1</td>
</tr>
<tr>
<td>Session2</td>
<td>Exercise2</td>
<td>Time2</td>
<td>Points2</td>
<td>HR2</td>
<td>EDA2</td>
<td>M12</td>
<td>M22</td>
<td>HR%Var2</td>
<td>Arousal2</td>
</tr>
<tr>
<td>Session3</td>
<td>Exercise3</td>
<td>Time3</td>
<td>Points3</td>
<td>HR3</td>
<td>EDA3</td>
<td>M13</td>
<td>M23</td>
<td>HR%Var3</td>
<td>Arousal3</td>
</tr>
<tr>
<td>Session4</td>
<td>Exercise4</td>
<td>Time4</td>
<td>Points4</td>
<td>HR4</td>
<td>EDA4</td>
<td>M14</td>
<td>M24</td>
<td>HR%Var4</td>
<td>Arousal4</td>
</tr>
<tr>
<td>Session5</td>
<td>Exercise5</td>
<td>Time5</td>
<td>Points5</td>
<td>HR5</td>
<td>EDA5</td>
<td>M15</td>
<td>M25</td>
<td>HR%Var5</td>
<td>Arousal5</td>
</tr>
<tr>
<td>Session6</td>
<td>Exercise6</td>
<td>Time6</td>
<td>Points6</td>
<td>HR6</td>
<td>EDA6</td>
<td>M16</td>
<td>M26</td>
<td>HR%Var6</td>
<td>Arousal6</td>
</tr>
<tr>
<td>Session7</td>
<td>Exercise7</td>
<td>Time7</td>
<td>Points7</td>
<td>HR7</td>
<td>EDA7</td>
<td>M17</td>
<td>M27</td>
<td>HR%Var7</td>
<td>Arousal7</td>
</tr>
<tr>
<td>Session8</td>
<td>Exercise8</td>
<td>Time8</td>
<td>Points8</td>
<td>HR8</td>
<td>EDA8</td>
<td>M18</td>
<td>M28</td>
<td>HR%Var8</td>
<td>Arousal8</td>
</tr>
<tr>
<td>Session9</td>
<td>Exercise9</td>
<td>Time9</td>
<td>Points9</td>
<td>HR9</td>
<td>EDA9</td>
<td>M19</td>
<td>M29</td>
<td>HR%Var9</td>
<td>Arousal9</td>
</tr>
<tr>
<td>Session10</td>
<td>Exercise10</td>
<td>Time10</td>
<td>Points10</td>
<td>HR10</td>
<td>EDA10</td>
<td>M110</td>
<td>M210</td>
<td>HR%Var10</td>
<td>Arousal10</td>
</tr>
<tr>
<td>AverageHR</td>
<td>AverageEDA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4.2.2.A Data Collection

In order to keep a record of the relevant data being generated from players’ activity and from EmoRegulators itself, a logging utility module for the game was designed and implemented. In earlier iterations of this work, we were focused on registering all the raw data we could collect from EmoRegulators without further calculations. This consisted of seven features: session, exercise and points gained, which are representative of the game state; and heart rate, EDA, bicep activation and trapeze activation, which constitute the raw physiological data that help define player state. The logging module would generate two types of logs: one which is meant to be read by humans, with proper indentation of the information and readable text and numbers, called the interaction log; the other one consisted of a nearly illegible log, called the feature log, formatted in such a way that it could be directly imported to Sci-kit Learn [23] (introduced further in this document) as a data set, meaning information such as the session
and exercise was transformed to follow a numerical coding, effectively discretizing these as features.

As the work objectives were revisited, we decided to add time as a feature as well. We implemented a timer system in EmoRegulators which, at any moment, would give us the time elapsed since the game session started. In the later months of this work, the idea of following both a subject-oriented approach and an exercise-oriented approach to data logging arose - but more on that later. This change in requirements was met with the implementation of additional timers, each measuring the time elapsed for each exercise that composes the game. Along with this change came the necessity to log exercise data separately, resulting in the creation of 13 additional logs - one for each exercise in the game - so that later on in training we could import this data to sci-kit learn in an easier way.

The logging is performed in real-time, meaning every half second the data corresponding to that moment is written to the appropriate log, as to avoid problems with data loss in case of unexpected crashes. Each logging instance is accompanied by the corresponding strategy performed: if no interaction request came from the WoZ interface in that instant, an inaction would be logged for that data; otherwise, the corresponding interaction would be logged with that data instead.

One additional change was performed to the way we collected data. Later on, we concluded that an observation containing a single sample collection of heart rate, muscle activations and so on could result in very poor model performance for several reasons. First of all, we would be allowing for outliers like observations with sensor reading errors (caused by sudden movement of test subjects, for example) to be part of the data set mixed up with other, actually relevant, observations with accurate readings - which would make it impossible for the model to pick up these outliers as noise. Moreover, we would be missing out on the connection that exists between subsequent values of heart rate or EDA, for example. Such a simple approach would inevitably result in loss of information that could improve model performance. For this reason, we thought that including multiple collected samples in the same observation would be a good step towards improving this. We decided to include 10 samples per observation since that was the same number of samples that was being shown for heart rate and EDA to the wizard during demonstrations. We couldn’t afford to to include too many samples into one observation since doing this reduces the overall number of observations in the data sets and we are already working with relatively small amounts of data. This decision is what made possible the calculation of the more complex features previously described, such as the averages for biometric data for example.

With observations having ten readings of heart rate, this meant there were ten measurements of percent variance in each observation (and likewise for arousal). The decision behind this is that algorithms could pick up on the relevance of such features that represent whether heart rate values “trend” upwards or downwards in any given moment of a session or exercise. This is particularly interesting in the sense that it could help detect moments of increasing or decreasing physical activity, as well as arousal tendencies in the wake of certain instructions or external speech acts, for example.
4.2.3 Restricted-Perception WoZ Interface

In order to teach the system using a restricted-perception Wizard-of-Oz paradigm, some means of ensuring the perceptions of the expert were controlled and restricted had to be implemented. Two steps are taken to achieve this. The first one, is that the expert would be interacting with EmoRegulators by using a carefully designed interface. This interface could not provide any more information than the one the system would use during training or runtime classification (these phases are explained in detail further in this chapter). This interface is shown in Figure 4.3 and focuses on the following:

i. Displaying information regarding the bio-sensors’ readings (i.e., the basic player state features)
ii. Showing the current context of EmoRegulators (i.e. the game state features)
iii. Mapping the calculated arousal value in a 0 to 10 scale, called the arousal meter
iii. Displaying a set of buttons that corresponding to the emotion regulation strategies the expert can choose from for the facilitator to say.

The second step was to conduct the restricted-perception WoZ demonstrations by ensuring the expert would be controlling the agent from a physically separate room than the one where the player would be interacting with the game. This was to guarantee that the expert would not make decisions by picking up on body language, facial expressions or other behavior while the game session took place.

The "About..." button seen in Figure 4.3 does not provide any additional perceptions or anything that would compromise the restricted-perception paradigm; it only displays information that credits the authors of the original icons used to design the ones seen in the buttons. A special detail that had to be taken into consideration was the number of bins, or values, being shown in the heart rate and EDA charts (in red and blue, respectively). Originally 30 values were shown at any instant in the interface, which had to be corrected to no more than 10 values, as to ensure this was consistent with the temporal window of 10 values (over 5 seconds) present in each observation that was being logged during game sessions.

Finally, it was concluded that if the audio played in the game wasn’t reproduced in the interface as well, the human expert using the interface would not be able to effectively predict a user’s emotional state. This change was accommodated as requested by the person playing the role of the expert in the earlier demonstrations in order to provide high-level context of what was happening in the game session at each instant. Note that, in any case, the expert was not watching the player or anything besides what is displayed on the interface; only listening to the dialogues.

When developing the interface, a number of decisions had to be made in regards to all the data traveling from the game to the interface and vice-versa. The most important one which has to be mentioned is the decision to register and send sensor data every 0.5 seconds. We feel this allows the expert to make an informed decision fast enough that it is still relevant when the user is presented the interaction resulting from that decision. While the technology used (such as the sensors and its software) didn’t
limit our ability to update data in even shorter time intervals, due to our initial choice of conducting the WoZ demonstrations with a specialist based in Italy - that is, having the specialist control the facilitator using the interface from Italy while having players test this version of EmoRegulators in Portugal - we couldn’t go lower than that due to latency (the communication delay) aggravation between the interface and the game.

There were also some changes that had to be made in respect to the audio management for the dialogue system of EmoRegulators, with the addition of the new speech utterances. In earlier iterations of this work's implementation, all the interactions would interrupt the assistant's speech act if there was one being performed. This was a result of how the model of EmoRegulators was implemented, where at the end of most speech acts, a component of the architecture called Session Manager would manage the process of transitioning to the next instructional segment, allowing for the previous segment to be skipped with the press of a button. Later we discussed this and after consulting with domain experts, we found that it should be possible to perform interactions freely, at any given moment in an exercise, regardless of whether an instruction was taking place or not. This change would allow for the expert to act upon changes in biometric data immediately without skipping instructions or having to wait until the end of an instructional segment - here we refer to instructional segments as the constituent parts of an exercise, where each exercise may have multiple segments with instructions and/or information regarding the exercise - which should obviously prove much more effective in addressing the emotional state of a player than otherwise.
The solution we found (which required actively notifying the players of it before the demonstrations started), was to allow for overlapping audio of two files, lowering the audio of any on-going instruction as the emotion regulation strategy audio was being played. A far from ideal solution, but the best compromise we came up with at the time given the limitations we found; and even then we still came across several limitations in audio libraries written for C# and .NET too, although we were able to find workarounds for specific issues with those (which are not worth going into detail).

4.2.3.A Design decisions

The interface shown in Figure 4.3 can be divided into three sets of elements. The first one informs the expert about the current game state, by displaying textual information regarding the session, exercise and number of points obtained thus far. There are other things we could have included, such as the time occurred since the beginning of an exercise or session, but ultimately we decided to keep only the information regarding the game state since different players take different amounts of time to complete each task. Not only is there nothing to be derived from such data, but it could even be harmful in that the expert could feel obligated to take this variable into consideration, potentially clouding their judgment.

The second one is the set of possible interactions which one can choose from. These are distinctively split into two sub-groups of motivational and empathetic strategies - not necessarily because they are different or may affect gameplay differently, but just to ease distinction between two for the interface user (the specialist), perhaps avoiding choosing the wrong interaction by accident. These buttons are large enough as to attempt to prevent this kind of mistakes in user input. We also consulted with a professor and researcher in the fields of Data Visualization and Human-Computer Interaction to validate an initial selection of the icons to accompany the name of the description in the buttons. Then we iterated further based on this feedback to create a new set of icons and we created a small questionnaire and asked researchers in this area to match the icons to the interaction descriptions. The icons in the questionnaire had the exact same pixel size as the ones designed for use on the interface, to ensure that feedback was as accurate as possible. Using a Pearson Chi-Square test to measure statistical association between the icons and expressions, we obtained a value of $p = 0.006 < 0.05$ thus supporting the usage of this version of the icons, which are the ones found in the current version of the interface.

The third set is dedicated to sensor data. There are two charts, one showing heart rate data and the other one EDA. They have different colors; the HR one is red, a color widely associated to the heart, and the EDA one is blue, another strong and distinct color from red, despite resembling dermal activity or not. There are also two switches representing whether muscle activity is “on” or “off” (that is, if it is occurring or not) for the trapeze and bicep muscles. Additionally, we decided to include an arousal meter, which attempts to measure arousal in a scale of 1 to 10 based on previous EmoRegulators sessions’ data.
Figure 4.4: A graphical representation of the emotion regulation loop during WoZ demonstrations.

The way the interface is meant to be used is quite straightforward, as you can see in Figure 4.4. Every half second, a request for updating the information is made from the interface to the game, and upon receiving a response, the new data is shown in the interface, updating it. The expert reads the information displayed in the interface and makes an assumption of what the player’s emotional state might be. Then, in case the expert thinks they should intervene, he/she selects the most suitable emotion regulation strategy and clicks the corresponding button. A request for that interaction is sent to the game via TCP and then it is mapped to its corresponding audio file to be played. The game sends a response as to confirm the requested audio will be played, and plays the audio. Upon receiving a confirmation response from EmoRegulators, the audio is reproduced on the interface side as well, adding to the expert’s perception of the game state. In case no interaction is deemed necessary, data flows between the interface and the game through update requests every half second as per usual.

4.2.4 Data Preparation

As we’ve seen previously in section 4.3.2, one of our implementation decisions was to collect data from the game every 0.5 second interval. And as stated before, the strategy being activated in the game at every time interval is recorded: if there’s no strategy being requested from the interface, an inaction is logged; otherwise, one of the strategies (whichever one was selected) is logged. What is bound to happen is that by the end of the full session, or even just one exercise, there will be a lot more samples recorded as inactions than any of the strategies, resulting in an imbalanced data set.
The problem with imbalanced data sets is that sometimes what one wants to learn the most happens the least - this is the case for the possible strategies being studied too. If one has a data set of 100 samples where 95 are inactions and the other 5 are interactions, how is the model ever going to learn when to perform the least occurring interactions? Even worse, what if some interaction (or label) does not occur once in a certain data set?

4.2.4. A Oversampling

![Diagram](https://www.marcoaltini.com/blog/dealing-with-imbalanced-data-undersampling-oversampling-and-proper-cross-validation)

**Figure 4.5:** A graphical explanation of how one should not perform oversampling in a cross-validation scenario. Image is taken from https://www.marcoaltini.com/blog/dealing-with-imbalanced-data-undersampling-oversampling-and-proper-cross-validation

One technique that can be used to address the first problem of imbalanced classes is called oversampling or upsampling. The idea is that we define some class weight for all possible classes in an attempt to balance the data set, allowing for all classes to have enough instances that the algorithm can learn when to perform every one of them. The oversampling algorithm will then generate synthetic data points that should only be marginally different from the real ones that are used to generate them. Of course, so can the opposite be done, and one could simply remove data points from the most occurring class or classes to bring the data set to a desired balance - this process is in turn called undersampling or downsampling.

In our case it would seem logical to perform undersampling of the inaction class, since the other classes have similar occurrences (between themselves) in different scenarios; however, when there is little data to work with, one should choose oversampling of the under-represented classes instead of undersampling by discarding real samples of the most represented class.
For this reason, we chose to perform oversampling of the under-represented classes in all our data sets. Knowing that realistically we will always have an excess of inactions compared to the eight possible interactions to choose from, we experimented with a 50% representation of inactions and 50% combined representation of the remaining classes. This means that each of those eight classes are still far less likely to be chosen over not perform an interaction, which we found was a reasonable compromise. We used a library with tools for imbalanced sets called imbalanced-learn [53] (with full support for sci-kit learn), which provides different algorithms for performing oversampling.

A question arises, though, when performing oversampling of the minority classes in a cross-validation scenario (we discuss cross validation in the training section further down). If we were to oversample before cross-validating, this could result in overfitting - despite cross-validation being a solution to avoid this problem. So how does it work? Marco Altini, a researcher in the field of health care and sports applications who applies machine learning methods in this field, explains this issue in a blog post [54]. Taking a simple oversampling scenario, he shows that by oversampling before performing cross-validation, we could end up with repetitions of the same sample which could then be split and be found in both the training and validation set, which as he says, defeats the purpose of cross-validation by testing on a sample which we trained our model with. This issue is represented in Figure 4.5, taken from the blog post. From the original data set, we oversample the minority class instances shown in yellow and orange and, by only later cross-validating, can end up testing our model on a training set which contains that same sample; meaning we're obviously testing the model with data that has been seen before, a typical approach that often leads to overfitting. If we choose to perform oversampling inside the cross-validation loop i.e. after splitting data into training and validation sets, as shown in Figure 4.6, we are making sure that no synthetic data points fall in the validation set.

As for the problem with classes that have no representation in a data set i.e. zero observations labeled as one particular class, we don’t believe there is any good solution for this. Nor would this be something we should fix: if across all subjects, a particular exercise does not have a single instance of any one label, then maybe it means that the emotion regulation strategy corresponding to that label shouldn’t be applied in that exercise. Another possibility would be, for example, that no labels corresponding to a remark on low motivation may actually mean that a player demonstrated lack of motivation in the whole duration of the EmoRegulators game session. What if the problem is the emotion regulation strategy itself, which may have never proven useful to begin with? For these reasons, it should be clear that even if we could, we should not create such instances of labels "out of thin air" - if the class has no representation, there is likely a reason for it, one which shouldn’t be ignored.

In previous sections we have shown you the confusion matrix and discussed multiple performance estimation metrics. We have also found in overfitting the main issue that storms and curses machine
learning models and algorithms. We’ve also seen in oversampling a solution to imbalanced data sets. In the following subsection we present the threefold approach we took for the training phase, inspired in another popular technique used to tackle that very problem.

4.2.5 Training Phase

In order to carry out the training phase of our solution, we primarily sought out a suitable machine learning library which we could use. We decided to use Sci-Kit Learn [55], an open-source machine learning library for Python. It features a very well-documented API and state-of-the-art implementations of a wide variety of algorithms, which we deemed a solid choice given that we were looking to compare several classifiers to decide which works the best specifically for EmoRegulators.

In the following section we discuss our three-fold approach to help determine the best model for EmoRegulators. These approaches were progressively chosen as a way to minimize the likelihood of problems like overfitting, introduced earlier, and take advantage of the pre-processing steps taken in data preparation before. These approaches were tested with the five algorithms we introduced in the Theory and Background, as to later choose the most appropriate and best performing algorithm for EmoRegulators based on the data collected in the WoZ demonstrations.
4.2.5. A k-Fold Cross Validation

In k-fold cross validation, the idea is to split the whole data set into any $k$ number of folds of equal (or near equal) size. Then $k$ iterations of training and validation are performed with a different fold being held out each time to be used as validation while the remaining $k - 1$ folds are used for training, as is depicted in Figure 4.7. A complementary method called stratification is often used prior to splitting the data into folds. This method consists of rearranging data to guarantee that each fold holds a class balance similar to the full data set. In EmoRegulators, this would mean ensuring that whatever the weighs of the different possible strategies (including inaction) are in the whole data set, each individual partition holds similar weights for each of its contained interactions as well. This technique is thus referred to as stratified k-fold cross validation.

The usage of k-fold cross validation with prior stratification in itself is already a powerful procedure to get reliable performance estimations, which is why, for the purpose of making an initial assessment to see how algorithms could perform using restricted perception WoZ studies, we decided to implement this variation of cross-validation. As for which value of $k$ to pick, many scientific sources [56–58] typically recommend using a value of $k = 5$ or $k = 10$ folds as a good sweet spot. For this reason, we chose a value of $k = 10$ to perform 10-fold cross validation.

For this assessment of our models’ performance, we merged all the subject-oriented data into a single dataset and then performed stratified k-fold cross validation. This data set is the one that gave us a testbed for performance estimation for different models on EmoRegulators, based on the features we chose to study. In the next two sections we discuss a different approach we took, where we intended to study subject-specific generalization and exercise-specific generalization, respectively.
4.2.5.B Leave-One-Subject-Out

We’ve seen repeated k-fold cross validation as a way to do model selection by estimating the accuracy of the different models we chose for comparison for EmoRegulators. What this approach gives us is a sort of estimation testbed to check how well a model could perform on EmoRegulators using restricted perception as we’ve just seen. However, we thought it would be interesting to also evaluate how these models generalize to new subjects specifically.

In our demonstrations we had subjects ranging from ages 14 to age 18, both male and female. We know from research that different people may have different baseline heart rates, for example. It is known that people who practice sports or engage in physically demanding activities tend to have a lower resting heart rate than others. Moreover, different people have different heart rate recovery times i.e. different people take different times to have their heart rate back to normal after performing some physically-intensive activity.

Given how heart rate plays such an important role in EmoRegulators - by allowing players to see their heart rate as they progress in the game, as well as in data collection for this thesis, for example - it makes sense that we want to explore a subject-oriented approach to the model’s generalization capabilities.

The idea behind this approach is very similar to how k-fold cross validation works, although it does not come from the same fundamental assumptions. Leave-One-Subject-Out, as we are calling it, is a term that is already used in machine learning to represent such an approach; however, unlike the
case of EmoRegulators, the term is sometimes used in contexts where a subject is represented by a single observation in a data set. In our case, subject-specific data is all the data gathered from a user’s demonstration session in EmoRegulators, and that constitutes a single data set. Obviously it’s a much bigger amount of samples, since each game session lasts about 45 minutes and we collect observations representative of 5 seconds of data each.

As such, for this approach we imported each subject’s data logs separately and stored them as different data sets. We can refer to Figure 4.8 to understand how Leave-One-Subject-Out was conducted; the idea is that each player’s data set acts as a fold. Here is where the first fundamental assumption about k-fold cross validation is not true for this approach: since different players took different times to finish the tasks in EmoRegulators, this results in some players’ data sets being bigger than others. Therefore, folds don’t necessarily have equal or near-equal sizes, as depicted in the figure.

Afterwards, the process seen in k-fold cross validation takes place: each of the $k$ iterations consists of using the $k^{th}$ player dataset as a test set, and using the remaining $k - 1$ player folds as the training set. The process is repeated $k$ times to allow a full rotation of test sets without any two test sets overlapping.

By analyzing accuracy with this approach, we can estimate how well a model can predict which actions to choose from in EmoRegulators when faced with unseen data - in this case, a possibly different subject, with likewise different heart rate and muscle behavior, as well as emotional traits and particularities affecting their emotional state which other players may or may not possess.

Just as it is interesting to look at how the model might generalize for new subjects, we additionally decided to see how models would be able to generalize for different exercises in EmoRegulators. We describe this approach in the following subsection.

**4.2.5.C Leave-One-Exercise-Out**

In chapter 3 we introduce EmoRegulators as a serious game and discuss some of the exercises that compose EmoRegulators as a whole. Exercises range from intense physical activity, like the one where the player is asked to dance to a song, to mindfulness exercises such as breathing regulation, which require little physical effort. These exercises have distinct heart rate ranges as a result. Other exercises, such as one where a player is required to metaphorically act as a snail and alternate between hiding and coming out of their shell, focus on localized muscle activity (in this case, the trapeze).

The nature of these exercises and the way biometric data manifests across different ones can be so different and peculiar that it also makes sense to study model selection and their generalization capabilities for exercise-oriented data. For this reason, we created separate logs from the main ones, focusing on extracting data from each exercise. We discarded features such as the current session and
exercise and, for each player session, we generated different logs for each exercise.

Analogous to the former approach, we called this one Leave-One-Exercise-Out. After collecting all the different data, for each particular exercise, we merged all data logs for that exercise from all the demonstration subjects. By doing this we get, for example, a data set for Progressive Muscle Relaxation containing data for this exercise from all subjects. This process is depicted in Figure 4.9 for a generic number of subjects and exercises.

Similarly to what happens with Leave-One-Subject-Out, we use each exercise data set (after merging it across players) as if it were a fold in k-fold cross validation and iterate \( k \) times, rotating a single exercise fold as the test set each iteration. Just like in Leave-One-Subject-Out, not all folds will have the same number of samples, since some exercises are shorter than others in time length. Lastly, in the next section we describe how we integrated the data we pre-processed and trained in Python into EmoRegulators for runtime classification.

### 4.2.6 Runtime Classification

Research was conducted with the purpose of finding tools that could facilitate Python integration with Unity3D, the game and content creation engine that EmoRegulators was built on. This is because while there are options for machine learning libraries in C# (the language EmoRegulators was implemented with in Unity), these are not as rich, robust or cutting-edge (when it comes to state-of-the-art implemen-
Figure 4.10: The exercise-oriented approach. Just like Leave-One-Subject-Out, data from a whole exercise is rotatively left out as unseen data to achieve better generalization results.

When training (and evaluating) algorithms (at least) as ones developed in languages such as Python or R, the two most sought languages for machine learning and data science in recent years. After researching, we decided to develop Python scripts to implement a simple socket-based communication with EmoRegulators. The way this works is similar to the way the WoZ interface interacts with the game for the demonstrations phase:

i. the sockets connect over TCP using the local address and a designated port;

ii. the Python socket script requests an update of EmoRegulators data every half second;

iii. the server socket on EmoRegulators side responds with the raw data (session, exercise, points and biosensor data) plus the time elapsed information in that instant;

iv. after 10 requests (that constitute the 5 second temporal window we describe in the data collection section) the Python script calculates the more complex features such as HR percent variance and averages from the 10 samples of raw data and then builds an observation with all the features;

v. the generated observation is passed to a predictive function of the machine algorithm chosen during model selection (this prediction is of course based on the previously trained data), returning the predicted classification of that observation;

vi. the Python script matches that class to the appropriate action and sends a request to EmoRegulators to play the corresponding interaction in the game.

These 6 steps are continuously repeated to generate predictions autonomously until the game session ends. This automated procedure works for any of the proposed approaches - whether it’s 10-fold cross validation, Leave-One-Subject-Out or Leave-One-Exercise-Out - with minimal computational cost,
but works better for some models than others, as more complex machine learning algorithms can take longer to make a prediction. The choices made in this department are explained in the following results chapter.
Results

Contents

5.1 Restricted-perception WoZ demonstrations ........................................ 63
5.2 Model Selection .................................................................................. 66
To round off this work, in this section we discuss the obtained results. First we detail how the restricted-perception WoZ demonstrations took place and what feedback was gathered from players and, afterwards, we move on to discussing the performance results of the five chosen algorithms across the 10-fold cross validation, Leave-One-Subject-Out and Leave-One-Exercise-Out approaches and our final choice of which combination was used for runtime classification in EmoRegulators.

5.1 Restricted-perception WoZ demonstrations

Back in section 3.2.1 when we presented EmoRegulators, we mentioned that the choice of exercises and elements that went into making the game were inspired by an Israeli protocol called BEAR, which was tuned to be suitable for individual teenagers rather than groups of children. The focus is on adolescents whose age lies between 14 and 18 years old. Given this age range and how the demonstrations took place during school season, it was exceptionally hard to come up with a suitable schedule.

Regarding the choice of the expert, a few requirements should be met to find a suitable candidate. First of all, it had to be someone familiar with EmoRegulators itself. There is a range of such different exercises and activities in the game that monitoring biometric data during the length of a session becomes a far less overwhelming task for someone who knows the sequence of the exercises and what players are meant to do in each of them; especially with the help of the instructions of the facilitator which are being sent from the game to the interface. Moreover, someone with a good knowledge of the original BEAR protocol and what core features were carried over to this transformed version would be an added benefit.

For these reasons, the most suitable candidate to play the role of the wizard would be Professor Esther J. Schek, who is among the people responsible for the original adaptation of the BEAR protocol to be incorporated into EmoRegulators. During the original study, she was responsible for monitoring tests with several adolescents and therefore has a lot of insight on which emotions were predominantly felt by the test subjects during the sessions, making her a valuable asset for the WoZ demonstrations as well. After inviting the professor to act as the wizard in the demonstrations of this thesis, we were met with a positive response.

A consent form allowing for the usage and processing of the children’s biometric data had to be signed by one of their parents or their guardian. After finding a suitable schedule for both the expert and the adolescents, we only managed to perform demonstrations with 5 subjects. The demonstrations were carried out in two different days and the group consisted of 3 boys and 2 girls: 2 of the boys were 14 years old, and the other 3 subjects were 17 years old. Demonstrations followed a protocol which was defined to facilitate communication between the expert and the researcher, which is detailed in the
After all the data logs from the demonstrations were collected a few days later, it was unfortunately found that not all the data from these first demonstrations was being properly collected making it unusable, forcing a rescheduling of new demonstrations.

The problem that caused the data loss was easy to fix, but hard to identify. Unfortunately, Professor Esther J. Schek would not be available for further demonstrations during the remainder of the acceptable time we had left to perform new Wizard-of-Oz demonstrations. Among the other possibilities for the role of the expert, the researcher was the best choice, given their extensive knowledge of the system and the BEAR protocol and the fact that in such a short remaining schedule to hand in this work, the researcher couldn’t afford to be dependent on other parties’ availability any more.

The researcher carried out these demonstrations in the same way these were carried out by the first expert initially - following the same protocol that was established for the initial WoZ demonstrations, acting as both parties involved. Fortunately, it was possible to schedule a second phase of restricted-perception Wizard-of-Oz demonstrations slightly over a week from the previous ones. The problems of some data loss seen then were fixed and an additional five students were gathered to fulfill the role of players. This time there were another three boys and two girls. The boys were aged 14, 15 and 18, while the two girls were both 17 years old, allowing for a good sample age-wise, despite smaller than preferable.

5.1.1 WoZ Protocol

A protocol was defined between the wizard and the researcher, so that there would be a standardized procedure across different demonstrations and so that communication between the two would be clear and effective in case any errors or problems arose during demonstrations - whether those were related to the interface, the biosensors readings or any other issue. The document detailing the protocol also mentioned technical decisions regarding the implementation of the interface, such as not allowing for the same emotion regulation strategy to be requested and played twice without a facilitator instruction playing in between, for example.

![Connect to EmoRegulators](image)

**Figure 5.1:** The interface window prompt shown to the expert to input EmoRegulators IP address.

Before each demonstration, an explanation of EmoRegulators was provided. This consisted of an
overview of the exercises, where some additional tips were provided regarding particularities of certain exercises. This was to ensure that there was minimal room for confusion so that interrupting a demonstration could be avoided. Afterwards, we would set up the sensors for HR and the two for muscle activation. Finally, the IP address of the computer EmoRegulators would run on was sent to the expert, so they could type it in the interface dialog shown in Figure 5.1 to connect to the game. After a ready check from the wizard, the game session would begin, the expert would click “Connect” and we would start monitoring the demonstration. While the expert monitored the session from the interface, the researcher’s job was to keep an open communication channel with the former in case there was any problem on the expert’s end, while at the same time checking every few minutes if the test subject needed anything - this was done by constantly checking through a narrow glass behind the subject and computer, as to ensure minimal interference and not need to open the door, possibly startling the subject and breaking their focus on the task at hand. At the end of each demonstration, the researcher would warn the expert that the former would close the session, so the latter could do the same and await further updates. The researcher would then talk with the subject for about 5 minutes while removing the sensors, to get general feedback.

5.1.2 Participants’ Feedback

In our initial planning of the work, the idea was to segment this into a workflow consisting of three phases, the last one being a final round of tests with players where they would be split into a control group, interacting with the original scripted version of EmoRegulators, and a condition group, interacting with this more interactive, automated version of the game. We wanted to accommodate this additional phase not because we needed it necessarily to increase model performance, but mainly for feedback on user enjoyment and emotional state in both versions. Unfortunately with the WoZ demonstrations happening so late in schedule, we weren’t able to conduct this final comparison between those versions. What we did, though, was try to gather some informations in the demonstrations phase.

In the WoZ demonstrations, while removing the sensors at the end of each one, the researcher asked the subjects for feedback on the tests. This was done in the form of a free discussion, as this was only a data collection task. No particular questions were asked in order to not prompt specific responses. In the first round of demonstrations, the subjects reported an overall fun and enjoyable experience but generally expressed feeling bored in the beginning of the session. They mentioned this was because of a 5 minute video shown at the beginning where no interaction is made; its purpose is to measure the player’s resting heart rate to use that as a staple for the remainder of the session. They mentioned this was because of a 5 minute video shown at the beginning where no interaction is made; its purpose is to measure the player’s resting heart rate to use that as a staple for the remainder of the session.

In the second round of demonstrations conducted by the researcher as the wizard, two of the subjects reported that they had a tough time associating certain feelings to parts of their body, which is required in one of the exercises, even though the procedure is explained. However, although it is reinforced that it is
not necessary for players to actively feel those emotions at the moment of doing the association between those and their body parts, it is understandable that the introspection required to do this assessment may be difficult.

Another critique that was made in this second round was how some interactions (requested from the interface) would sometimes overlap with instructions. This is the intended behavior resulting from a few limitations which we discussed previously, and which we initially expected could be a criticized aspect of the dialogue system.

The demonstrations went smoothly with the exception of one of the players becoming stuck in an exercise; when he signaled, the time and number of observations in the log up until that moment was recorded, the player was assisted, and the moment in time when the demonstration was resumed was recorded as well, so that it was possible to later remove the samples collected during the intervention from the data logs that would later on be imported as data sets for the machine learning model selection.

Other than that, a couple subjects commented on the timing of the interactions, explaining that for the most part the interventions were properly timed, but not always reflective of their emotional state: one of them specifically mentioned an instance when the strategy that points to lack of motivation was used at a time when it didn’t reflect the player’s feelings towards the exercises or the game, criticizing the usage of that intervention. One of the girls also disliked the overlapping of audio: she said despite the explanation that it could happen, it was hard to understand the underlying audio of the instruction, although she added it didn’t affect her ability to carry out the task at hand. We acknowledged beforehand that this would likely be among the most criticized aspects of the experience and given more time we would have definitely have manually annotated the audio instructions as previously suggested; this is definitely something which should be fixed as future work on EmoRegulators. After finally having accurate and complete data from all the demonstrations, we were finally able to import these observations as data sets for the subsequent training phase.

### 5.2 Model Selection

In this section we present the performance results achieved by the aforementioned machine learning algorithms. We evaluated them in four metrics: accuracy, F1 score, precision and recall. We also present our choice for the model to perform runtime classification and discuss these results for 10-fold cross validation, and leave-one-subject-out and leave-one-exercise-out approaches.

Looking at the data from all approaches, one thing becomes immediately clear: kNN, decision tree and random forest all see consistently great performance. In all approaches, as expected, random forests have always at least marginally better accuracy and F1 scores than decision trees, which makes
sense given that a random forest in sci-kit predicts the class with highest mean probability estimate across all trees that compose it.

Table 5.1: Performance estimation in our 10-fold cross validation approach

<table>
<thead>
<tr>
<th></th>
<th>kNN</th>
<th>Decision Tree</th>
<th>Random Forest</th>
<th>Gaussian Naive Bayes</th>
<th>Neural Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>0.967</td>
<td>0.905</td>
<td>0.967</td>
<td>0.060</td>
<td>0.889</td>
</tr>
<tr>
<td>F1</td>
<td>0.967</td>
<td>0.907</td>
<td>0.967</td>
<td>0.060</td>
<td>0.924</td>
</tr>
<tr>
<td>Precision</td>
<td>0.967</td>
<td>0.906</td>
<td>0.967</td>
<td>0.060</td>
<td>0.913</td>
</tr>
<tr>
<td>Recall</td>
<td>0.967</td>
<td>0.908</td>
<td>0.967</td>
<td>0.060</td>
<td>0.931</td>
</tr>
</tbody>
</table>

The Neural Network’s performance is tricky to analyze. However, the algorithm tends to perform better with extensive and representative training sets [59]. Our results corroborate this: in 10-fold cross validation, where we train with 90% of the whole data and each fold is very representative of the full data, we achieve high scores across all metrics, as we can see in table 5.1; these scores decrease for Leave-One-Subject-Out, where each fold should constitute roughly a fifth of all the data, making for a training set of around 80% of all data; in the exercise-oriented approach, where we train with 12 out of 13 folds, there is a slightly bigger training set than in 10-fold cross validation (considering the average between all 13 iterations), but the data in each fold is not guaranteed to be even slightly representative of the whole data set, which seems to justify its poor performance as well.

Table 5.2: Performance estimation for the Leave-One-Subject-Out approach.

<table>
<thead>
<tr>
<th></th>
<th>kNN</th>
<th>Decision Tree</th>
<th>Random Forest</th>
<th>Gaussian Naive Bayes</th>
<th>Neural Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>0.887</td>
<td>0.956</td>
<td>0.967</td>
<td>0.082</td>
<td>0.363</td>
</tr>
<tr>
<td>F1</td>
<td>0.909</td>
<td>0.945</td>
<td>0.950</td>
<td>0.134</td>
<td>0.490</td>
</tr>
<tr>
<td>Precision</td>
<td>0.933</td>
<td>0.935</td>
<td>0.935</td>
<td>0.932</td>
<td>0.930</td>
</tr>
<tr>
<td>Recall</td>
<td>0.887</td>
<td>0.956</td>
<td>0.967</td>
<td>0.082</td>
<td>0.363</td>
</tr>
</tbody>
</table>

Unsurprisingly, Gaussian Naive Bayes didn’t have a good accuracy performance for Leave-One-Subject-Out either. This time, though, it reported a precision score of 0.932 (see table 5.2). What this means is that across all classes (or all emotion regulation strategies), in 93.2% of the cases, the corresponding strategies were correctly identified; there were almost no false positives identified. Despite its high number of hits, however, pairing this with its extremely low recall score, we understand that many type II errors occurred. It was very “hit or miss”, which is in accordance with what we’ve seen about this classifier in chapter 2: it is a bad estimator in classification tasks. This behavior persists in the exercise-oriented approach, summed in table 5.3, but with a lower precision score at 44.7%, which seems to suggest this classifier’s precision for the data in EmoRegulators decreases with increasingly bigger differences in train/test set ratios. Its poor performance across all metrics and approaches made it clear this was not a viable algorithm to use in EmoRegulators for runtime classification.

kNN also seems to perform very well across all approaches generally good, with the biggest gap performance-wise between it and other top performing classifiers (Decision Tree and Random Forest)

67
Table 5.3: Performance estimation for the Leave-One-Exercise-Out approach.

<table>
<thead>
<tr>
<th>Method</th>
<th>kNN</th>
<th>Decision Tree</th>
<th>Random Forest</th>
<th>Gaussian Naive Bayes</th>
<th>Neural Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>0.908</td>
<td>0.964</td>
<td>0.972</td>
<td>0.040</td>
<td>0.790</td>
</tr>
<tr>
<td>F1</td>
<td>0.927</td>
<td>0.954</td>
<td>0.958</td>
<td>0.060</td>
<td>0.856</td>
</tr>
<tr>
<td>Precision</td>
<td>0.947</td>
<td>0.945</td>
<td>0.945</td>
<td>0.447</td>
<td>0.954</td>
</tr>
<tr>
<td>Recall</td>
<td>0.908</td>
<td>0.964</td>
<td>0.972</td>
<td>0.040</td>
<td>0.790</td>
</tr>
</tbody>
</table>

occurring for Leave-One-Subject-Out; its slightly lower performance suggests that a lot of type II errors are made, that is, a lot of classes are inappropriately being labeled as inactions when they shouldn’t. This is especially bad in EmoRegulators, where there is already a substantially bigger number of samples labeled as inactions than there labeled as the classes that are most interesting to us: the ones corresponding to the eight emotion regulation strategies we implemented according to the aforementioned emotion regulation strategies. Given that one of our secondary objectives is to keep a player engaged and with a positive attitude while playing EmoRegulators, these slight decreases in recall scores are especially hindering our ability to achieve this. For this reason, when selecting the model to be used during runtime classification, we chose to rule out k-Nearest Neighbors.

Analogously, despite promising results for 10-fold cross validation the Neural Network has a fairly low performance in terms of accuracy, F1 score and recall in Leave-One-Subject-Out. The reason we point its poor performance in this particular approach to ruling out MLP from model selection leads us to another discussion: which approach should we be using? Better yet, with with approaches’ training data do we want to do runtime classification in EmoRegulators?

Note that earlier we talked about progressively choosing these three approaches. What this meant is that rather than coming up with these three approaches concurrently, there was a sequential order to their implementations and inclusion in our studies. We began with 10-fold cross validation as a way of analyzing how the system performed across the different classifiers; it was essentially a testbed for looking at some of the aforementioned problems like overfitting, the fact that we had a small amount of samples and also to learn about oversampling and undersampling. Nevertheless, we wanted to evaluate the different models performance for this approach as well, which is why it is still included in this analysis. At this point, it was known that we should use the trained data from the Leave-One-Subject-Out approach, as the ultimate goal is to have good generalization results for players that are new to EmoRegulators.

As for Leave-One-Exercise-Out, it was a posterior decision to include it as an experimental method. We still believe the exercise-oriented approach is an interesting path to walk when we look mainly at how EmoRegulators as a game could be further improved and complemented with new exercises or activities. However, no changes to the game were made during this work in terms of what exercises there are. Generalizing for new exercises is therefore not of the utmost importance for what we set out to achieve in this work. What we initially sought to do was to come up with emotion regulation strategies...
that might combat feelings of boredom and frustration, keep the player engaged, and provide a better sense of interactivity - and these emotions and notions change from player to player.

While these results certainly looked promising, in doing this analysis we became curious as to which of the interactions or respective strategies actually worked better. We knew that there was a big problem with data imbalance, which we can refer to in table 5.4, which we attempted to solve by applying oversampling.

Table 5.4: Absolute frequencies of each class in our data set. Class 0 corresponds to inaction, while the remaining 8 correspond to ER strategies and respective interactions.

<table>
<thead>
<tr>
<th>Class</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>1955</td>
<td>6</td>
<td>5</td>
<td>14</td>
<td>11</td>
<td>9</td>
<td>8</td>
<td>11</td>
<td>3</td>
</tr>
</tbody>
</table>

When producing the results shown in tables 5.1, 5.2 and 5.3, we had to choose a value for an averaging parameter that determines how averaging of the precision, recall and F1 scores of all classes is made in sci-kit learn. The value we chose for this parameter was "weighed", since it was the only one which stated it accounts for label imbalance, which appeared to be the right value for our case. However, in order to look at the values for each class specifically, we changed this value to None; this is when we realized that our aforementioned results were highly inflated by, or biased towards, inactions, as we can gather from the results in table 5.5 which displays recall scores for each class individually across the five machine learning algorithms chosen for training.

Table 5.5: Recall scores of each class across different models for the Leave-One-Subject-Out approach.

<table>
<thead>
<tr>
<th>Class</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.91</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.22</td>
<td>0</td>
<td>0.3</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Decision Tree</td>
<td>0.87</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.99</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Gaussian NB</td>
<td>0.05</td>
<td>0.1</td>
<td>0.1</td>
<td>0</td>
<td>0</td>
<td>0.6</td>
<td>0.2</td>
<td>0.1</td>
<td>0.11</td>
</tr>
<tr>
<td>Neural Network</td>
<td>0.63</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.2</td>
<td>0.29</td>
<td>0.5</td>
<td>0</td>
<td>0.17</td>
</tr>
</tbody>
</table>

The main takeaway at this point was that several of the interactions were not even being properly learned; class 3 in particular, which corresponds to the "Proud Of You!" interaction, can't by learned by any of the chose classifiers. Because we're interested in learning the strategies and hopefully sense their impact in gameplay, we had to come up with some approach that could improve performance - even if only slightly - especially for recall, which is the performance metric we're most interested in improving, while maintaining an acceptable accuracy given the circumstances.

In order to achieve this, we took it a step further and decided to merge interactions derived from each strategy into a group consisting of that strategy, i.e., interactions based off of the attribution theory into one group, then the ones based on cognitive reappraisal into another, and the remaining empathetic and motivational remarks into a third group. The idea behind this was to increase the number of occurrences
of each class, i.e., their frequency in the data set, in an attempt to ease learning of the data for the various models chosen. Our expectation was to therefore achieve improvements in recall for Attribution Theory, Cognitive Reappraisal and Empathetic/Motivational strategy groups at the expense of the recall score of Inactions, in order to improve interactivity and diversification in gameplay.

<table>
<thead>
<tr>
<th></th>
<th>Inaction</th>
<th>Attribution Theory</th>
<th>Cognitive Reappraisal</th>
<th>Empathetic/Motivational</th>
<th>Accuracy of Classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>kNN</td>
<td>0.91</td>
<td>0</td>
<td>0.25</td>
<td>0</td>
<td>0.88</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>0.71</td>
<td>0.5</td>
<td>0.25</td>
<td>0.02</td>
<td>0.69</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.85</td>
<td>0</td>
<td>0.1</td>
<td>0</td>
<td>0.82</td>
</tr>
<tr>
<td>Gaussian NB</td>
<td>0.05</td>
<td>0.2</td>
<td>0.7</td>
<td>0.12</td>
<td>0.06</td>
</tr>
<tr>
<td>Neural Network</td>
<td>0.58</td>
<td>0.14</td>
<td>0.44</td>
<td>0.08</td>
<td>0.57</td>
</tr>
</tbody>
</table>

After this merging of strategies, we arrived at the results seen in table 5.6. As we can see, classifiers are better at learning these strategies now that each class has a slightly larger number of observations. In particular, we found that Decision Tree, Gaussian Naive Bayes and Neural Network were able to learn all strategies, even if at low recall scores. We can also see that Cognitive Reappraisal in particular is learned by all classifiers. Although these results are far from ideal, this merging strategy was the best solution we could come up with after this dissertation was done, given that it was not possible to conduct more demonstrations to obtain a significant amount of data to work with. Nevertheless, we arrived at a fairly good compromise: Decision Tree is able to learn all strategies with the lowest decrease in accuracy, which makes it our choice of the classifier to be used when performing real time classification in EmoRegulators. While kNN and Random Forest have better accuracy, neither saw an increase in recall score for Attribution Theory and Empathetic/Motivational strategies, which is what we were looking to achieve. As for Gaussian Naive Bayes and Neural Network, both were fairly below expectations in terms of their predictive accuracy, which is why we valued Decision Tree higher - although, to be fair, in the presence of a significantly large and representative data set, as we’ve discussed, Neural Network could potentially be the better choice for classification.

Despite the poor results in classification for the eight singular interactions we designed for this work, we achieved a considerably fair compromise for better results by applying this strategy unification. Furthermore, we can conclude that Leave-One-Subject-Out is the most interesting training approach to take in the context of EmoRegulators because of the inherent generalization capabilities, given our particular emphasis in providing a more meaningful experience for different and new players in EmoRegulators. Finally, we believe these poor results are a consequence of the tiny number of meaningful samples (the ones corresponding to the occurrence of the designed interactions) we collected in our demonstrations; these results should not in any way invalidate the usage of a restricted-perception WoZ approach. Further research has to be conducted to assess the viability of this method with a significantly larger data
set.
Contents

6.1 Conclusions .................................................. 75
6.2 System Limitations and Future Work .......................... 76
6.1 Conclusions

As their lives become increasingly more dependent on technology every day, humans are finding ways to push their boundaries by pushing technology itself to new heights across numerous fields such as health care and education. In the latter, as the student to teacher ratio becomes larger and larger, so have learning solutions to accommodate this change started to increase in demand and popularity. A lot of research is being done in fields pertaining to embodied pedagogical agents and tutoring systems. In this work, we sought out to apply a restricted-perception Wizard-of-Oz paradigm to EmoRegulators, a serious game that teaches players self-regulation, by providing individualized support based on a automated model that learned from human input, in an effort to make an in-game instructional agent into an intelligent tutor.

In our solution we discuss that the problem of how we can create an intelligent and adaptive system to help maintain engagement in EmoRegulators can be split into two subproblems, and we present a solution that uses emotion regulation strategies to attempt to improve the gameplay experience of EmoRegulators through a WoZ paradigm. We discuss our model, where we detailed how data was collected and prepared for training, as well as the thought process behind building the interface for the demonstration phase. These demonstrations, conducted with 5 adolescents between ages 14 and 18, allowed us to gather data that was then used for training with the five algorithms and a three-fold approach, allowing us to conduct a classifier comparison to decide which combination to use in runtime classification, made possible by using TCP socket communication to bridge EmoRegulators in Unity and the scripts for training and runtime scripts written in Python.

Finally, we discussed results for the culmination of the work conducted in this thesis. Through a small free and informal talk in demonstrations, we learned that players reported an overall positive experience, but also expressed concerns regarding the dialogue system of the in-game assistant - an implementation decision that needs a pondered revision. After a deeper look at our model performances, we decided to take a different direction and group each strategies' interactions together as a way to ease learning for the models. We saw a slight improvement in recall at a reasonable enough sacrifice in accuracy in the case of Decision Tree, making this our choice for real time classification. Our initial takeaway that Leave-One-Subject-Out is the most relevant training approach to take stands, as it improves the model's capability of generalizing for new players, which is something we value highly for EmoRegulators. A small number of demonstrations held looks to be the culprit of the less than ideal results we arrived at, and we believe this should not take away any value in conduction restricted-perception WoZ studies - this method's viability in EmoRegulators needs to be further researched before taking any objective conclusions.
6.2 System Limitations and Future Work

Looking back, there are a few interesting directions to go from what we've learned. Conducting demonstrations with such a small sample of players didn't allow us to properly assess the solution's performance for a representative amount of player profiles, habits and physiological tendencies and behavior. Going forward, many more demonstrations should be conducted before training, preferably with a more suitable expert like Professor Esther J. Schek, for example. More data would also allow us to properly test more complex and interesting models such as neural networks without these handicaps.

Given the time and opportunity, the previously mentioned third phase of comparing both the neutral and the affect-aware versions of EmoRegulators would have been conducted, with different players assigned to different groups for each of those versions. This would allow to collect deeper feedback on the system and self-reported levels of enjoyment and emotions through questionnaires, effectively assessing whether we have or haven't created an intelligent, adaptive system in EmoRegulators, capable of being aware of, understanding and regulating players’ emotions during an interaction session. The dialogue system should also get the revision it deserves: a proper annotation of all speech acts and external interactions, or a dialogue system capable of performing seamless or human-like discourse transitions and interruptions.

Finally, it could be beneficial for other systems if some of the conducted research and work done were put towards building an actual framework for conducting Wizard-of-Oz studies to build affect-aware versions of existing systems, although this would require a more profound revision of the higher-level requirements and more extensive research on other existing views on emotion regulation and strategies.
Bibliography


[23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and


Interface Icon Matching Questionnaire

The following is the questionnaire that was formulated to ask experts in the fields of Information Visualization to exclusively match icons to the emotion regulation strategies employed in this thesis.
User Feedback Questionnaire – Icons and Expressions

*Required

1. Please choose for which of the following expressions the icon is better suited. The expressions aren't listed in any particular order. (Por favor escolha para qual das seguintes expressões o ícone apresentado mais se adequa. As expressões não estão listadas por nenhuma ordem em particular.) *

   Mark only one oval.
   
   - So many points!
   - You’ll feel better!
   - Talking too much
   - Proud of you!
   - Good job!
   - Long Exercise
   - Almost there!
   - You can do it!

Matching icons and expressions

2. Please choose for which of the following expressions the icon is better suited. The expressions aren't listed in any particular order. (Por favor escolha para qual das seguintes expressões o ícone apresentado mais se adequa. As expressões não estão listadas por nenhuma ordem em particular.) *

   Mark only one oval.
   
   - So many points!
   - You’ll feel better!
   - Talking too much
   - Proud of you!
   - Good job!
   - Long Exercise
   - Almost there!
   - You can do it!

Matching icons and expressions
3. Please choose for which of the following expressions the icon is better suited. The expressions aren't listed in any particular order. (Por favor escolha para qual das seguintes expressões o ícone apresentado mais se adequa. As expressões não estão listadas por ordem em particular.) *

Mark only one oval.

- So many points!
- You’ll feel better!
- Talking too much
- Proud of you!
- Good job!
- Long Exercise
- Almost there!
- You can do it!

Matching icons and expressions

4. Please choose for which of the following expressions the icon is better suited. The expressions aren't listed in any particular order. (Por favor escolha para qual das seguintes expressões o ícone apresentado mais se adequa. As expressões não estão listadas por ordem em particular.) *

Mark only one oval.

- So many points!
- You’ll feel better!
- Talking too much
- Proud of you!
- Good job!
- Long Exercise
- Almost there!
- You can do it!

Matching icons and expressions
5. Please choose for which of the following expressions the icon is better suited. The expressions aren't listed in any particular order. (Por favor escolha para qual das seguintes expressões o ícone apresentado mais se adequa. As expressões não estão listadas por nenhuma ordem em particular.) *

Mark only one oval.

- So many points!
- You'll feel better!
- Talking too much
- Proud of you!
- Good job!
- Long Exercise
- Almost there!
- You can do it!

Matching icons and expressions

6. Please choose for which of the following expressions the icon is better suited. The expressions aren't listed in any particular order. (Por favor escolha para qual das seguintes expressões o ícone apresentado mais se adequa. As expressões não estão listadas por nenhuma ordem em particular.) *

Mark only one oval.

- So many points!
- You'll feel better!
- Talking too much
- Proud of you!
- Good job!
- Long Exercise
- Almost there!
- You can do it!

Matching icons and expressions
7. Please choose for which of the following expressions the icon is better suited. The expressions aren't listed in any particular order. (Por favor escolha para qual das seguintes expressões o ícone apresentado mais se adequa. As expressões não estão listadas por nenhuma ordem em particular.) *

Mark only one oval.

- So many points!
- You'll feel better!
- Talking too much
- Proud of you!
- Good job!
- Long Exercise
- Almost there!
- You can do it!

Matching icons and expressions

8. Please choose for which of the following expressions the icon is better suited. The expressions aren't listed in any particular order. (Por favor escolha para qual das seguintes expressões o ícone apresentado mais se adequa. As expressões não estão listadas por nenhuma ordem em particular.) *

Mark only one oval.

- So many points!
- You'll feel better!
- Talking too much
- Proud of you!
- Good job!
- Long Exercise
- Almost there!
- You can do it!
WoZ Demonstration Consent Form

The consent form that had to be filled by the adolescent’s parents or tutor consenting usage of their biometric data for the WoZ demonstrations.
Autorização

Eu, encarregado(a) de educação de ___________________________________________,
autorizo a sua participação em testes com utilizadores para uma tese de mestrado no
Instituto Superior Técnico no campus do Taguspark, e dou o meu consentimento para que
lhe sejam recolhidos dados biométricos (batimento cardíaco e activação muscular) durante
a realização dos testes. Os dados permanecerão anónimos e não serão utilizados fora do
contexto desta tese.

___ de Março de 2018,

____________________