



# DSIE'20

Faculty of Engineering  
University of Porto  
Porto | Portugal

Proceedings  
of the  
15th Doctoral Symposium  
in  
Informatics Engineering

Editors:

A. Augusto de Sousa

Carlos Soares

11-12 February 2020

<https://fe.up.pt/dsie2020>

# FOREWORD

## STEERING COMMITTEE

DSIE - Doctoral Symposium in Informatics Engineering, now in its 15th Edition, is an opportunity for the PhD students of ProDEI (Doctoral Program in Informatics Engineering of FEUP) to show up and prove they are ready for starting their respective thesis work.

DSIE is a series of meetings that started in the first edition of ProDEI, in the scholar year 2005/06; its main goal has always been to provide a forum for discussion on, and demonstration of, the practical application of a variety of scientific and technological research issues, particularly in the context of information technology, computer science, and computer engineering. DSIE Symposium comes out as a natural conclusion of a mandatory ProDEI course called "Methodologies for Scientific Research" (MSR), leading to a formal assessment of the PhD students first year's learned competencies on those methodologies.

The above mentioned specific course (MSR) aims at giving students the opportunity to learn the processes, methodologies and best practices related to scientific research, particularly in the referred areas, as well as to improve their own capability to produce adequate scientific texts. With a mixed-format based on a few theory lessons on the meaning of a scientific approach to knowledge, practical works on analyzing and discussing published papers, together with multidisciplinary seminars, the course culminates with the realization of a meeting/symposium. Then, DSIE may be understood as a kind of laboratory test for the concepts learned by students. In this scope, students are expected to simultaneously play different roles, such as authors of the submitted articles, members of both organization and scientific committees, and reviewers, duly guided by senior lecturers and professors.

DSIE event is seen as an opportunity for the students to be exposed to all facets of a scientific meeting associated with relevant research activities in the above-mentioned areas. Although still at an embryonic stage, and despite some of the papers still lack maturity, we already can find some interesting research work or promising perspectives about future work in the students' thesis. At this moment, it is not yet essential, nor often possible, for most of the students in the first semester of their PhD, to produce sound and deep research results. However, we hope that the basic requirements for publishing an acceptable scientific paper have been fulfilled.

Each year DSIE Proceedings include papers addressing different topics according to the current students' interest in Informatics. This year, the tendency is on Intelligent Systems (7 papers), followed by Robotics and Simulation (2 papers) and Software Engineering (2 papers).

The complete DSIE'20 meeting lasts one day and a half and includes one invited talk by an academic researcher, Professor Alípio Jorge, from the Faculty of Sciences, University of Porto. Professors responsible for ProDEI program's current edition are proud to participate in DSIE'20 meeting and would like to acknowledge all the students who have been deeply involved in the success of this event. Hopefully, this involvement contributes to their better understanding of the themes addressed

during the MSR course, the best scientific research methods and the good practices for writing scientific papers and conveying novel ideas.

Porto, February of 2020

Carlos Soares and A. Augusto de Sousa  
(Steering Committee of DSIE 2020)

# FOREWORD

## ORGANIZING AND SCIENTIFIC COMMITTEES

The chairs of the Organizing and Scientific Committees of the Doctoral Symposium in Informatics Engineering (DSIE'20) warmly welcome you to the DSIE 15th edition. With great honor, we have accepted the invitation to be a part of these committees. Organizing an event, like the DSIE, confirmed to be both a challenging and practical task, in which all the people involved have certainly derived great value.

The joint effort of our colleagues from the Doctoral Programme in Informatics Engineering (ProDEI), was fundamental in making this event a success. We believe that these efforts are reflected in the quality of this year's DSIE.

Our first acknowledgment goes to our supervisors, Professor Augusto Sousa and Professor Carlos Soares. We would like to thank them for their time and efforts in making this conference possible and for providing us with all the invaluable concepts. A special thanks to Professor Alipio Jorge for accepting our invitation as Keynote Speaker.

We would like to thank all the senior members of the Scientific Committee for their involvement, the junior members for their collaboration, and the invaluable support of Sandra Reis and Pedro Silva from the Informatics Engineering Department (DEI) from the Faculty of Engineering of the University of Porto (FEUP). Finally, we would like to thank our sponsors because their help undoubtedly reflected on the quality of DSIE'20.

And, above all, we thank you, our authors and other participants, for being a part of DSIE'20!

Porto, February 2020

Liliana Patrícia Saldanha Antão (Scientific Committee Chairs)

Inês Filipa Nunes Teixeira (Organization Committee Chair)

Sara Filipa Couto Fernandes (Organization Committee Chair)

# CONFERENCE COMMITTEES

## **STEERING COMMITTEE**

A. Augusto Sousa  
Carlos Soares

## **ORGANIZING COMMITTEE CO-CHAIRS**

Sara Fernandes  
Inês Nunes Teixeira

## **ORGANIZING COMMITTEE**

Ahmed Fares  
Beatriz Cruz  
Carlos Tomás  
Daniel Garrido  
Eduardo Ribeiro  
Eliseu Pereira  
Filipa Ivars  
José Aleixo  
Liliana Antão

## **SCIENTIFIC COMMITTEE CHAIR**

Liliana Antão

## **SENIOR SCIENTIFIC COMMITTEE**

Ademar Aguiar  
Américo Pereira  
Arnaldo Pereira  
Carlos Soares  
Carla Teixeira Lopes  
Daniel Silva  
Eduardo Almeida  
Filipe Correia  
Gil Gonçalves  
Henrique Lopes Cardoso  
Hugo Sereno Ferreira  
Jaime Cardoso  
João Jacob  
João Moreira  
João Reis  
Luís Teixeira  
Luís Vilaça  
Luiz Paulo Pires  
Mário Antunes  
Paula Raissa Silva  
Paula Viana  
Pedro Campos

Rosaldo Rossetti  
Rui Camacho  
Rui Maranhão Abreu  
Rui Nóbrega  
Rui Pinto  
Tiago Soares da Costa  
Zahid Iqbal

**JUNIOR SCIENTIFIC COMMITTEE**

Ahmed Fares  
Beatriz Cruz  
Carlos Tomás  
Daniel Garrido  
Eduardo Ribeiro  
Eliseu Pereira  
Filipa Ivars  
Inês Nunes Teixeira  
José Aleixo  
Sara Fernandes

# SPONSORS

DSIE'20 – Doctoral Symposium in Informatics Engineering is sponsored by:

## Main Sponsors:

**PRODEI** PROGRAMA DOUTORAL EM ENGENHARIA INFORMÁTICA

**DEI** DEPARTAMENTO DE ENGENHARIA INFORMÁTICA



## Gold Sponsors:



## Silver Sponsors:



A Flutter company

## Bronze Sponsors:



# CONTENTS

<b>Invited Speaker</b>	
Alípio Jorge .....	9
<b>Session 1 - Software Engineering</b>	<b>10</b>
Can Safety-Critical Systems Engineering adopt Agile? Why not? <i>Eduardo Ribeiro</i> .....	11
Live Metrics Visualization: An approach to help software developers <i>Sara Fernandes</i> .....	12
<b>Session 2 - Intelligent Systems I</b>	<b>13</b>
Predicting Predawn Leaf Water Potential up to seven days using Machine Learning <i>Ahmed Fares</i> .....	14
Fault Injection, Detection, and Treatment in Simulated Autonomous Vehicles <i>Daniel Garrido</i> .....	15
Affective Computing: Concepts Analysis, Review and Future Directions <i>Filipa Ivars</i> .....	16
Analyzing the Credit-Assignment Problem in Deep Reinforcement Learning <i>José Aleixo</i> .....	24
<b>Session 3 - Robotics and Simulation</b>	<b>32</b>
Teaching a Robot Precision Ball Sports using Deep Reinforcement Learning <i>Liliana Antão</i> .....	33
Realistic wheelchair simulator using gazebo and ROS <i>Ana Beatriz Cruz, Armando Sousa, and Luís Paulo Reis</i> .....	34
<b>Session 4 - Intelligent Systems II</b>	<b>41</b>
MSDS-OPP: Operator Procedures Prediction in Material Safety Data Sheets <i>Eliseu Pereira</i> .....	42
Offensive assessment in social networks <i>Carlos Tomás</i> .....	50
Finding Emotions in Screenplays for Genre Classification <i>Inês Nunes Teixeira</i> .....	56
<b>Poster Session</b>	<b>62</b>
Chain Fusion: A novel fusion method in multimodal learning <i>João Barbosa, Bernardo Ramos, Pedro Azevedo, Ricardo Silva, and Alejandro     Gudino</i> .....	63
Wi-Fi-based network systems design over freshwater: Experimental evaluation using COTS devices <i>Miguel Gutiérrez Gaitán, Pedro M. Santos, Luis R. Pinto, and Luis Almeida</i> .....	64



# INVITED SPEAKER

Professor Alípio Jorge

*“What is really boosting AI?”*

Alípio Jorge works in the areas of data mining, machine learning, recommender systems, and NLP. He is a graduate in Applied Mathematics/Computer Science by UP, a PhD in Computer Science, also by UP, and MSc by the Imperial College.

He is an Associate Professor of the Department of Computer Science of the University of Porto since 2009 and is the head of that department since 2017. He is a researcher and the coordinator of LIAAD, an associated unit of INESC TEC. He has projects in web automation, recommender systems, information retrieval, text mining and decision support for the management of public transport.

He lectures Machine Learning and Programming. He coordinated the MSc in Computer Science from 2010 to 2013. He was part of the team that launched the Masters on Data Science which he coordinated from 2017 to 2019. He collaborates with the MAP-i PhD program.

As a lecturer at the Faculty of Economics from 1996 to 2009, he launched with other colleagues the MSc in Data Analysis and Decision Support Systems. He was in charge of this master since its creation, in 2000, until 2008. He represents Portugal in the Working Group on Artificial Intelligence at the European Commission and is the coordinator for the Portuguese Strategy on Artificial Intelligence, "AI Portugal 2030".

# SESSION 1

## Software Engineering

Can Safety-Critical Systems Engineering adopt Agile? Why not?

*Eduardo Ribeiro*

Live Metrics Visualization: An approach to help software developers

*Sara Fernandes*

# Can Safety-Critical Systems Engineering adopt Agile?

J. Eduardo Ferreira Ribeiro  
Faculty of Engineering, University of Porto  
Department of Informatics Engineering  
Porto, Portugal  
jose.eduardo.ribeiro@fe.up.pt

**Abstract**—For many years safety-critical systems domains are heavily regulated. To confirm the quality and safety of components and systems as a whole, companies working in these environments need to follow the standards that describe in detail the phases of the development life cycle and techniques to confirm the quality and safety of components and systems as a whole. Following these standards is what will make organisations successfully achieve the certification of their products. Due to the heavily regulated environment and the need to achieve successful certification, safety-critical systems companies have been doing their development by using the traditional waterfall model. Traditional waterfall model has been used to manage the development processes. While this is still in use, the creation of agile manifesto and its methods, provided an alternative to organisations looking to improve their work. The scope of the present research was to gain an understanding of whether organisations within safety-critical systems engineering can adopt the agile methodology. The objective was to identify relevant publications by conducting a literature review of articles published in Scopus and Inspec databases. The current findings show us that there is work being done in the area. Nevertheless, there is space for future work. In the conclusions, we identified some point for future work that can be followed through

# Live Metrics Visualization: An approach to help software developers

Sara Filipa Couto Fernandes  
Faculty of Engineering, University of Porto  
Rua Dr. Roberto Frias, 4200-465 Porto, Portugal  
up201405955@fe.up.pt

*Abstract*—Software systems are becoming more complex, making their comprehension and maintainability more complicated, costly, and time-consuming. Sometimes, it is necessary to restructure a program before changing it, refactoring it, improving not only its structure but also its quality. There are already several tools that check the quality of a software system. Still, these tools don't provide immediate feedback about the system to its developers. To mitigate this problem, we propose the development of a plugin that provides feedback, in real-time, of several software metrics that help developers to know the state and quality of their software. Besides, this tool also allows the recommendation of two refactorings and the comparison of the metrics with the ones from different Git versions. To validate this tool, we carried out a controlled experiment, where the participants had to improve a project, with and without the created tool. We did different hypothesis-tests using the results obtained. However, we could not reject our null hypotheses due to several validation threats. Thus, through this research, we were able to develop a tool that gives immediate feedback to developers on the state of their system, but it still needs some validation improvements.

# SESSION 2

## Intelligent Systems I

Predicting Predawn Leaf Water Potential up to seven days using Machine Learning

*Ahmed Fares*

Fault Injection, Detection and Treatment in Simulated Autonomous Vehicles

*Daniel Garrido*

Affective Computing: Concepts Analysis, Review and Future Directions

*Filipa Ivars*

Analyzing the Credit-Assignment Problem in Deep Reinforcement Learning

*José Aleixo*

# Predicting Predawn Leaf Water Potential up to seven days using Machine Learning

Ahmed Fares

Faculty of Engineering, University of Porto

LIAAD-INESC TEC, Porto, Portugal

ahmed.a.fares@inesctec.pt

**Abstract**—Sustainable agricultural production requires a controlled usage of resources such as water, nutrients, and minerals from the environment. Different strategies of plant irrigation are being studied to control the quantity and quality balance of the fruits. Regarding efficient irrigation, particularly in deficit irrigation strategies, it is essential to act according to the status of water stress in the plant. For example, in vine, to improve the quality of the grapes, the plants are deprived of water until they reach particular water stress before re-watered in specified phenological stages. The water status inside the plant is estimated by measuring either the Leaf Potential during the Predawn, or soil water potential, along with the root zones. Measuring soil water potential has the advantage of being independent of diurnal atmospheric variations. However, this method has a lot of logistic problems, making it very hard to apply along all the yard, especially the big ones. In this study, the Predawn Leaf Water Potential (PLWP) is daily predicted by Machine Learning models using data such as grapes variety, soil characteristics, irrigation schedule, and meteorological data. The benefits of these techniques are the reduction of the manual work of measuring PLWP and the capacity to implement those models on a larger scale, by predicting PLWP up to 7 days which should enhance the ability to optimize the irrigation plan while the quantity and quality of the crop is under control.

# Fault Injection, Detection and Treatment in Simulated Autonomous Vehicles

Daniel Garrido

Faculty of Engineering, University of Porto

Rua Dr. Roberto Frias

4200-465 Porto, Portugal

up201403060@fe.up.pt

***Abstract***—In the last few years autonomous vehicles have been on the rise. This increase in popularity lead by new technology advancements and availability to the regular consumer has put them in a position where safety must now be a top priority. With the objective to increase the reliability and safety of these vehicles, a fault detection and treatment modules for autonomous vehicles were developed for an existing multi-agent platform that coordinates them to perform high-level missions. Additionally, a fault injection tool was also developed to facilitate the study of said modules alongside a fault categorization system to help the treatment module select the best course of action. The results obtained show the potential of the developed work, with it being able to detect all the injected faults during the tests in a small enough time frame to be able to adequately treat these faults.

# Affective Computing Towards Nonverbal Communication

Filipa Ivars Silva, Rosaldo J. F. Rossetti

*Artificial Intelligence and Computer Science Laboratory (LIACC)*

*Faculty of Engineering of University of Porto (FEUP)*

Porto, Portugal

up200906162@fe.up.pt, rossetti@fe.up.pt

**Abstract**—Affective Computing can be interpreted as computing that recognizes and simulates emotions. This study focuses on simulating emotions resorting to certain Psychology foundations that relate emotions to colors; however, this relationship varies from one's culture, age, personal experience, among other factors. Therefore, we propose a solution with a proof of concept developed in this study that simulates emotion through color, learning the user's unique emotion-color profile. The system obtains this profile by learning from the user through Human-Computer Interaction, and which the learning process is based on Reinforcement Learning, in particular, the Q-Learning algorithm. We explore the role of empathy in Affective Computing and Human-Computer Interaction scopes with a focus on the emotion simulation pillar. We suggest that communication between humans and computers can be established through emotion, and our results showed that the solution we created is able to learn a user's emotion-color dictionary within an average of 270 interactions with the user and to mimic the user's emotion once the profile is learned. To sustain our hypothesis, we briefly analyze some aspects of Psychology regarding emotions and its correlation with Affective Computing research focus.

**Index Terms**—Affective computing, color psychology, emotion, empathy, human-computer interaction (HCI), reinforcement learning, Q-learning

## I. INTRODUCTION

Affective Computing has had an increasing interest as a single area of study but also combined with other areas such as Human-Computer Interaction (HCI) and Artificial Intelligence (AI). Throughout the years, new combinations of Affective Computing with other fields are being discovered followed by new applications.

This field of research was formally introduced in 1995 as "computing that relates to, arises, or influences emotions" [1]. Since then, new measures of emotions and new applications have been discovered promoting its popularity.

Since Picard first reference to Affective Computing, it was introduced to HCI as a tool to improve, since it would be possible to measure the user appraisal in run-time by the interaction with a product or system and, therefore, it was suggested that the product could also adapt in run-time to the user experience [2].

Along with technological evolution, Artificial Intelligence emerged, and its relation with AC has grown stronger also, mainly for Artificial Agents, in which emotion recognition has shown as a good resource to interact with the user [10] [17]

[24]. AI has an important role in Affective Computing as well since the recognition of emotions requires the computer to learn emotion indicator patterns in user's facial expressions or even in the speech [12] [18] [25].

We begin by interpreting in a simplistic perspective Picard's definition of Affective Computing as "computing to recognize and simulate emotion", and this work will focus on the emotion simulation. Our solution addresses the hypothesis that it is effective to use emotions as an interaction method between human and computer through a nonverbal communication model based on emotion-color relationship.

Humans can naturally relate colors to emotions; however, this relationship varies with some external factors such as culture, gender, age, and personal experiences [3]. Artificial Intelligence role in this work is to enable our solution to learn what is the user's emotion-color profile, or dictionary.

We applied Reinforcement Learning to build a Q-Learning model in a client-server solution which interacts with the user using a visual interface that returns to the user a color that represents the emotions provided by him or her and, once the user gives a feedback to inform the solution whether the color represents his or her emotions state or if it does not.

This work contribution is relevant for Affective Computing literature because it explores the capability of a computer to simulate emotion in view of the fact that most of the efforts of previous work concentrate, especially in emotion recognition. Nonetheless, it also contributes to Human-Computer Interaction publications considering that it explores multi-modal interaction through emotions and color and communication through a nonverbal language based on each particular user profile. Additionally, it contributes to the Artificial Intelligence field with the application of a Q-Learning model which considers how confident is the system that learned the user's emotion-color profile.

Lastly, the literature has several references combining Affective Computing with Human-Computer Interaction and with Artificial Intelligent separately, however, is worth to mention that, in this work, we try to combine the three fields in order to assemble a solution that explores them evenly.

The potential applications are vast, but we tried to suit this solution to be adequate to systems that don't require the user's full attention removing detected emotion outliers considering only the emotions in which the user returns a



'yes' or 'no' feedback. Intelligent vehicles are artificial agents and companions are suitable for applications for the reasons described and, also, solutions for autistic patients since they are described as good pattern detectors and whose interaction with colors as showed good results while they struggle with emotion recognition and simulation [17].

The structure of this article is composed of a review of previous works, procedures, and methods approached in this study, description, and details of the proposed solution to test the formulated hypothesis, presentation and discussion of the results of the experiment conducted and, finally, the conclusion of this study.

## II. RELATED WORK

Along with the growing interest and the proven capacity of Affective Computing to be combined with several research fields, there are many in ongoing research and others that are in its early experiments.

We present a compilation of a set of the most significant fields of research combined with Affective Computing in the following subsections.

### A. *Affective Computing and Psychology*

Affective Computing is deeply connected to emotions and, therefore, to Sentiment Analysis. We can define emotion as a state composed of three characteristics: a subjective experience, a physiological response, and a behavioral or expressive response. Moreover, Sentiment Analysis is a computational study focused on opinions, sentiments, emotions, and attitudes expressed [24].

Empathy is also worth to mention since it "is a set of cognitive and affective skills we use to make sense of and navigate the social world" [8]. It is a process composed by three sub-processes: 1) an emotional simulation process to mirrors the other's emotional elements; 2) a perspective-taking process to understand other's emotions; 3) an emotion regulation process to appease other's pain or discomfort leading to a compassionate and helping behavior [9].

Picard mentioned emotional intelligence in one of her initials publications regarding AC, as very important to human adaptation to its environment and evolution [4]. In order to improve this adaptation, solutions to address issues such as market prediction, artificial companions for autistic children, depression monitoring, and artificial pedagogical agents are growing in popularity within academic literature [8] [15] [17] [24] [27].

Yet, there is yet a subject to be addressed that arises from psychology to since Picard questioned it in 2001: what is the nature of emotions in human's body so that the differentiation between emotions is achievable and what factors does this differentiation of emotions depend on [5].

Society has been suffering from some aspects of economic growth. Children are presenting signals of social capacities deficiency due to extreme loneliness leading to psychological and health issues. Some researches are being conducted for applications of Affective Computing to address this issue.

Autistic patients are also a target of this subject research due to their lack of human-human interaction capacities [17].

Companion and tutor pedagogical agents are emerging to aid both professor and student in the learning experience by complementing knowledge contents, learning content, and also to provide emotional support. A project that serves as an example is *DIMI Agent* [10]. Yet, other studies suggest that there is a need of Affective Computing systems to be more integrated with the online learning platform since it is an emerging way of learning and has tools to provide those systems the affective information needed [27].

1) *Color Psychology*: The relation between emotion and color has long been discussed, and previous works suggest that this relation is usually natural for humans but varies from person to person. Despite this variation in the emotional perception of color, there are some rules and trends that also determine this perception caused by general assumptions of gender colors and advertising. In previous works, some researchers tried to purpose mathematical models to represent the effects of factors on color-emotion responses [22].

Elliot stated in 2014 that "color is about more than aesthetics—it can carry important information and can have an important influence on people's affect, cognition, and behavior.". His work underlines that color is directly related to attention and perception and interpersonal communication [16].

Previous work also suggests that "Music-color associations are mediated by emotion.". Experiments showed a strong relationship between facial expressions and color, music and color, and emotions to music even though they were not able to obtain a conclusion about cross-cultural variations [14].

### B. *Affective Computing and Human-Computer Interaction*

The mention of the relationship between Affective Computing and Human-Computer Interaction was first stated when Picard presented AC concept in 1995 [1].

In a more generic perspective, AC is indicated as a tool for HCI to address some human interaction issues with computers, such as a) reducing user's frustration; b) enabling communication of user's emotion; c) building tools to help develop social-emotional skills; d) developing solutions to handle the affective information [2].

Nevertheless, Picard's idealized applications of Affective Computing to HCI, even in its early stages, are not yet met. In 2004, Muller expressed concern about the lack of knowledge from HCI fields to study some of the questions proposed by Picard, suggesting that the researchers should conduct further research to analyze how to integrate Affective Computing to other areas. The conflict between 'how' should it be used in user interaction with machines was stated by the comparison of interaction interpretation from Turkle and Picard by Muller, since the first referenced to computers as "tools to think with" while the latter referenced as "tools to feel with" [6].

### C. Affective Computing and Medical Applications

Medical applications for Affective Computing are being researched mainly for psychology assistance. Autism, depression, and stress are examples of use cases for Affective Computing. Autism is a condition that limits the ability to empathize with others. Systematizing empathy was suggested in previous works as a solution to help autistic patients to recognize and respond to other's emotions and, also, to computers to learn how to apply Affective Computing effectively [8] [23].

Other studies report that in cases of a) Alzheimer's disease; b) cardiovascular diseases in patients suffering from stress; c) mood diseases, including depression and bipolar disorder; d) anxiety disorders; e) anger management issues. Affective Computing can play an important role in helping to assess the obstacles they all have in common: emotion management, either ones' emotion or others'. It could be helpful to support Cognitive Behavioral Therapy [24].

### D. Affective Computing and Artificial Intelligence

As earlier mentioned, systematizing the process of emotional recognition requires pattern recognition, and new ways to provide it with the best performance are being discussed and researched. Artificial Intelligent has an important role in the techniques researched and still under research [12] [18]. Text classification through detected emotions is emerging due to the immense applications from author recognition to threat detection from social media. Affective Computing has been having an important role in this subject since it was first mentioned the measurement of emotions in speech recognition, and the similarity between the two was natural [11] [23].

Virtual agents that can communicate with the user by recognizing and expressing emotions is also included in the literature to address the issues discussed in the section regarding Psychology as a field of research of Affective Computing, and AI is a big part required to create these virtual agents [10] [17] [27].

Games with Artificial Intelligence in order to be able to make better decisions and interact with the user are also emerging with some components of Affective Computing [18] [21].

Ambient Intelligent focuses on forecasting the user's emotion by processing other users' emotions and the environment information. A study regarding this topic was conducted, and the conclusion was that their Affective Computing model is reliable but has less accuracy than the conventional methods. However, the model can identify features that were not able to be identified previously [26].

### E. Affect Measurement Tools

Facial expression is the most used measure of emotions in Affective Computing, but there are more, and the set is growing with the number of researches and interested researches from different areas.

The most six common measures in Affective Computing literature are: 1) facial expressions; 2) speech expressions; 3) text expressions; 4) body language; 5) physiological signals;

6) eye movement. Recent studies mentioned the Affective Haptics, whose goal is to understand how the sense of touch can recognize and express emotions [20]. Therefore, we can also consider the haptic expressions as a measure of Affective Computing.

Also, the head motion has been defended as an independent measure instead of being considered together with facial expressions, and it's mentioned that the information provided by head motion is complementary to the emotion content from facial expressions, rather than redundant [19].

And finally, eye-blink startle response is also being proposed as an Affective Computing measure to indicate surprise and fear as a response of audio stimulus [13].

In order to elicit emotions, the affective systems require that the collected emotions are properly classified and, sometimes, it is the user that provides the emotion that classifies them by self-report and concurrent expression. Self-reported emotions are provided by the user that immediately classifies it, interrupting the experience of collecting the emotion. In contrast, concurrent expression provides the user the liberty to continue the experience, and sensors and other tools elicit the emotions without the user intervention [4].

The emotions addressed nowadays have limitations since the indicators that distinguish emotions from each other are weaker with many emotions than otherwise - in which fewer emotions are likely to be differentiated by string indicators. The set of emotions most common in academic literature is Ekman's, and there are also other two well know from Arnold and Plutchik. Zucco [24] compiled these sets in order to be more natural to compare them in a table that is represented by Table I.

TABLE I  
AFFECTIVE COMPUTING EMOTION CATEGORIES

Author	Emotion Category
Ekman	anger, disgust, fear, joy, sadness, surprise
Arnold	anger, aversion, courage, dejection, desire, despair, fear, hate, hope, love, sadness
Plutchik	acceptance, anger, anticipation, disgust, joy, fear, sadness, surprise

## III. PROCEDURES AND METHODS

As a solution that aims for the computer to be able to simulate emotions by mimicking the user, it is required to interpret emotions, learn which is the user's profile emotion-color, and communicate with the user through the obtained dictionary. The applied methods are described in the following subsections.

Being capable of simulating the user's emotions using his or her unique profile, satisfies one of the three pillars of empathy. This solution is our method to test the present hypothesis of this paper by analyzing the effectiveness of the system to provide an environment for human-computer nonverbal communication.

Affective Computing, Human-Computer Interaction, and Arti-

ficial Intelligence, in particular, Reinforcement Learning, were approached to assemble the final solution.

#### A. Emotions as a Mean of Communication

As suggested by Elliot [9] in previous work, emotion simulation is one of the empathy components. The literature has several references regarding emotion interpretation in Affective Computing field; however, we tried to approach the other aspect: the simulation of emotions by a computer process.

Has the computer interprets the user emotions, it tries to learn from the user's feedback how to correlate those emotions into a language that can be both used and interpreted by the user and the computer.

Color has long been related to emotion in Psychology mainly to persuade the user's behavior, so we believe that its relationship with emotions could be used to mediate the interaction human-computer with emotions as a means of communication.

#### B. Emotion-Color Approach

Color interpretation varies from each individual, and so does its relationship with each of the individual's emotions. This variation depends on several factors such as culture, age, gender, and personal experiences.

Therefore the language based on an emotion-color dictionary can not be generalized for every user. It is required for the computer first to interpret the user's emotion to learn afterward what is the user-specific emotion-color profile by learning with the interaction with him or her.

#### C. The Solution Technical Design

The solution created has a final purpose of being able to simulate emotions by mimicking the user's emotions using color as an emotion representation.

It was required for the computer to be able to interpret emotions, an interface to express emotions to the user by colors and to collect the user's evaluation of whether the obtained emotion represents his or her emotional state, and the ability to learn from to user in order to conclude what is the user's emotion-color profile so that it can be used as a language to communicate.

The interactions with the user are presented in Fig. 1, and the interface is represented in Fig. 5. The interface is composed by a start button to initialize the emotion recognition and the communication between front-end and back-end layers processes, a stop button to interrupt these processes, and a 'yes' and a 'no' buttons to allow the user to provide the feedback regarding the color returned by the system.

1) *Technical Details and Methods:* Our system is composed by two main layers: 1) the front-end, in which the interface that captures the user feedback and shows the color is placed, and the emotion interpretation; 2) the back-end, where our Artificial Intelligent engine runs with Reinforcement Learning, in particular, with Q-Learning. This structure is presented in Fig. 3.

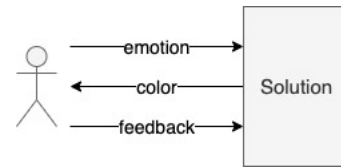


Fig. 1. User interaction with the solution.

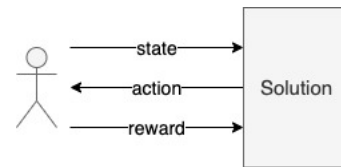


Fig. 2. User interaction with the solution considering Reinforcement Learning nomenclature.

The first component was built using *HTML* and *CSS* for the interface and *Javascript* for the emotion interpretation. As we explained earlier in this paper, our focus is on emotion simulation. Therefore we decided to use the *Affectiva API*<sup>1</sup> for emotion interpretation. *Affectiva API* is a product from the *Affectiva* company that grows out from MIT's Media Lab and is co-founded by Rosalind W. Picard, one of the pioneers of Affective Computing. We were given access to this API academic license provided by the company.

The software development technologies used were selected for simplification purposes considering we did not aim to create an appealing and elaborate visual solution. Therefore, product design methods were disregarded.

The back-end runs through a Python<sup>2</sup> script since it is widely used for scientific and numeric computing, where we implemented our Q-Learning model. The communication between the two layers is made through WebSockets, as a bidirectional communication channel, by Transmission Control Protocol (TCP) connection.

#### D. The Solution Functionality

1) *Interpreting Emotions:* Our research did not focus on interpreting emotions. Therefore we integrated the *Affectiva API* in our solution using a provided Software Development Kit (SDK). This API interprets emotion from the user's facial expressions identifying the face feature points and the position between them.

This technology also requires Artificial Intelligence so that it has the ability to detect the patterns of human facial expressions. The emotion detection technology that provides this learning ability is called *Emotion AI*<sup>3</sup>. Technical and scientific details about the applied methods to recognize emotions are not mentioned by the product provided.

<sup>1</sup>www.affectiva.com

<sup>2</sup>www.python.org

<sup>3</sup>www.affectiva.com/emotion-ai-overview

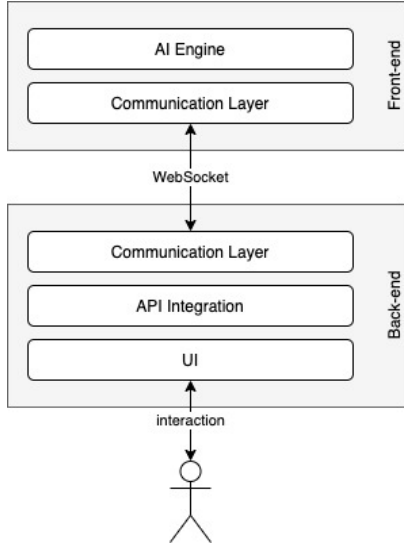


Fig. 3. The solution's technical architecture.

2) *Simulating Emotions*: In order to simulate emotion considering the user's emotion-color profile, the solution needs to be able to learn from the user. Once the user provides an emotion, the system will interpret it and categorize it as one of the following five emotions: 1) *neutral*, 2) *joy*, 3) *sadness*, 4) *anger*, and 5) *fear*. The system will return a color for the user to evaluate if it relates to his or her emotional state, and, finally, the user will provide a yes or no feedback according to the evaluation, allowing the solution to learn from the reward obtained from the obtained feedback. Fig. 4 represents the system's flow diagram figuring these interactions.

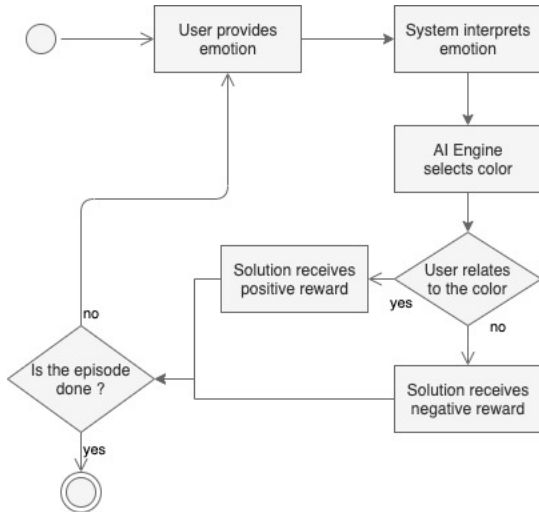


Fig. 4. The solution's flow diagram.

We are considering an emotion set (Fig. 6) that resembles the set of emotions categories proposed by Ekman in Zucco's

publication [24]. For simplicity purposes, we aggregated disgust and anger into anger category, and surprise and fear into fear due to the similarity between the two, since the indicator that allows the emotions recognition is based on the distance and position between the facial feature points.

The learning process is based on the Q-Learning algorithm owing to the simplicity of the solution interacting with the user, which is based on state-action-reward, as you can observe in Fig. 2, parallel to Fig. 1.

The Q-Learning algorithm was first introduced by Watkins in 1989. Its purpose is to learn the state-action value (Q-value), represented by  $Q(s, a)$ , in order to allow the system to decide the best action to achieve a given goal by calculating the optimal policy. The Q-values that correspond to each possible action for each state are mapped into a table - the Q-table. The standard procedure of the Q-Learning algorithm is given as follows [7]:

- 1) Randomly initialize the Q-values of the Q-table
- 2) Obtain the current state ( $s_t$ )
- 3) Following a certain policy, select an action ( $a$ ) for the given state
- 4) Execute the selected action
- 5) Receive the reward ( $r$ )
- 6) Perceive the next state ( $s_{t+1}$ )
- 7) Update the correspondent Q-value according to the equation 1
- 8)  $s_t \leftarrow s_{t+1}$
- 9) Go to step 3 until the episode-done condition is met
- 10) Repeat the steps 2 to 9 for a given number of episodes

An episode is the cycle from step 2 to 9.

Our Q-Learning model considers a learning rate of 0.1, and a discount factor of 0.95. It is based on the Bellman Equation represented by equation 1.

$$Q(s_t, a_t) \leftarrow (1 - \alpha) \cdot Q(s_t, a_t) + \alpha \cdot (r_t + \gamma \cdot \max_a Q(s_{t+1}, a)) \quad (1)$$

where:

- $\alpha$  - learning rate
- $\gamma$  - discount factor
- $s$  - state
- $a$  - action
- $r$  - reward
- $t$  - current iteration
- $t + 1$  - future iteration

In order to balance *exploration* and *exploitation*, we use the  $\epsilon$ -greedy algorithm by initializing the  $\epsilon$  as 1 and the  $\epsilon$  decays at a rate of 30% every 50 iterations.

Because emotions are perceived from the user through the computer's webcam, we observed a large oscillation of emotions being registered and outliers occurred. To prevent these outliers and improve emotion-color learning accuracy, we implemented an emotions average calculation system consisting of clustering emotions in a given time-frame,

which we configured to 5 minutes, and calculate the emotion that was most perceived. The emotion evaluation for the learning process is the emotion obtained from this calculation.

In each iteration, the reward value is used to update the Q-values of the Q-table from the model, following the equation 1. Our Q-table is represented by a matrix of emotions  $\times$  colors. Throughout an episode, multiple iterations occur, and each iteration corresponds to an interaction from the user providing feedback.

Nevertheless, a condition to evaluate whether the solution was able to learn or not during an episode, was required to complete the interaction flow in the learning process. As a result, we implemented an episode-done condition based on the standard deviation of the Q-values per row, i.e., per emotion.

Each time the Q-values are updated, as the solution learns, the Q-value of the color of each emotion that meets the user's emotion-color profile tends to increase while the other almost-equally decreases. Thus, the standard deviation increases.

Once the standard deviation of the five emotions reaches 0.9, the system considered the episode-done condition is met, and the episode ends with a reward value 0, meaning success. Otherwise, the reward is -1.

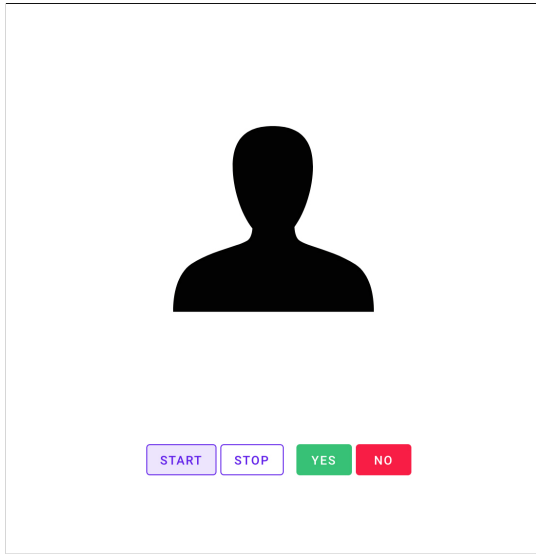


Fig. 5. The solution's interface model. A 'start' button to initialize the emotion recognition, a 'stop' button to interrupt the emotion recognition, and a 'yes' and a 'no' buttons to allow the user's feedback.

#### IV. RESULTS AND DISCUSSION

We set the time-frame in which the user's emotion average is calculated, to five minutes. We anticipated the need for a large number of user's interactions for the system to be able to learn the user's emotion-color profile. Therefore, we created a simulation environment to automatically emulate the user's interaction considering a given emotion-color profile

previously defined.

This simulation environment begins by filling an emotion-color first column matrix with random emotions from our emotions set. Afterward, we loaded the matrix's second column with the emotion's corresponding colors. Each row represents an interaction solution-user. Thus, this simulation environment communicates directly with the solution's back-end (Fig. 3) completing the state-action-reward cycle (Figs. 1 and 2).

##### A. Experimental Results

Our experiment consisted of 1000 rounds, and we verified that our solution was able to learn a user's emotion-color profile after an average of 270 interactions for two previously selected profiles (Fig. 6). There was no discernible difference in the results between the two given profiles. One can observe

NEUTRAL	NEUTRAL
JOY	JOY
SADNESS	SADNESS
ANGER	ANGER
FEAR	FEAR

Fig. 6. Emotion-color profiles used in the experiment by the simulation environment. Profile 1 at the left, and Profile 2 at the right.

in Fig. 7 the evolution of the average reward for iteration in a selected episode. An episode reaches its end once the system "is confident" that has found the user's emotion-color, accordingly to the previously detailed condition in Procedures and Methods section.

User tests weren't considered in this experiment.

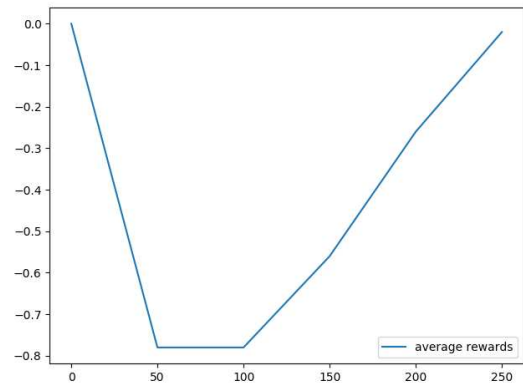


Fig. 7. Evolution of the average reward by iteration in an episode.

##### B. Discussion of Results

Considering the determined 5 minutes time-frame, and the results of the experiment obtained utilizing the simulation environment described, the system would require an average

of 22 hours and 30 minutes of consecutive interactions with the user to learn its emotion-color profile. But, since we didn't consider user tests, we are not able to conclude if this forecast behavior reflects the reality of real user experience.

The lack of conclusion due to the absence of user tests raises the question of the capacity of this solution to create a sufficiently strong engagement environment to maintain the user's interest and attention in order to enable the system's learning process until the user's emotion-color is found.

Additionally, the simulation environment didn't include errors while creating the matrix in which the experiment was based. Therefore the exploration of the Q-Learning model is considered straightforward leading us to question if, in a real environment, the system would be able to learn at the same rate as the experiment revealed.

The hypothesis of this study includes the possibility of the system being able to demonstrate empathy towards the user. The mimic approach to simulate the user's emotion was proven to be successful since the solution reflects the user's emotion through color considering the user's specific emotion-color dictionary. Therefore, we concluded that this solution serves as a proof of concept to further research.

## V. CONCLUSIONS

In this paper, we stated our hypothesis as "It is effective to use emotions as an interaction method between human and computer through a nonverbal communication model based on emotion-color relationship." This hypothesis was associated with empathy considering emotion simulation as one of the pillars.

Supported by literature references, we based our study on the relationship between emotions and colors in order to provide our system with a nonverbal language to interact with the user through color.

We created a solution that interprets emotion provided by the user through a camera, using the *Affectiva API*. The solution proved to be able to learn a user's emotion-color profile after an average of 270 interactions. The learning process is based on Reinforcement Learning, in particular on the Q-Learning algorithm. Therefore, we conclude that this study is to be considered as a successful proof of concept to a system that interacts with a user through emotions.

The limitations to highlight include the absence of user tests in our experiment and error injection in our experiment to simulate a real HCI environment better. It is worth to mention that the user's mean to provide the feedback so that the solution is able to learn, is verbal (yes and no buttons), whereas our hypothesis is based on nonverbal communication. However, we considered this fact as a minor limitation to our solution's definition.

This study contributes to the research fields of Affective Computing, Human-Computer Interaction, and Artificial Intelligence. The first mainly benefits from the proposed approach to use colors to simulate emotions. We assessed the contributions of the second by having as a foundation of this study the analysis of the effectiveness to use emotions in an

HCI closed system as means of communication, thus including empathy between humans and computers as an HCI factor to consider. And, finally, this study brings contributions to the third field with our model in which the Q-Learning algorithm was applied, and with our confidence model to allow the system to evaluate whether it was able, or not, to learn the user's emotion-color profile.

The solution provided by this study has potential to several applications of nowadays research fields, such as 1) Artificial Pedagogical Agents to support both teacher and study managing emotions; 2) Intelligent cars, to detect and prevent dangerous situations, like car accidents; 3) Internet of Things (IoT) solutions to facilitate interaction with the user; 4) Autistic patients to explore emotions interpretation through color pattern from interaction; 5) Artificial Personal Agents considering this solution's assumptions and results as a multi-modal interaction mean.


As further work recommendations, we highlight the benefits of performing user tests and analyze its results against the obtained results from the simulation environment used in this study. Also, literature references the importance of other color factors not considered, such as hue and chroma; therefore, include these in this study scope would bring additional relevance. We believe that comprising neural networks in this study would improve the current learning process contemplating some general known presumptions relating emotions to colors. Facial expression recognition was the only explored measure for the user's perceived emotions. Thus, exploring other Affective Computing measures would improve emotion recognition accuracy. The color was used as a representation of color; however, previous works suggest that this representation could also be accomplished through music. And, finally, we explored only one pillar of empathy, and we consider that it would be of great importance to explore the remained two in order to understand its role in Affective Computing and Human-Computer Interaction research fields.

## REFERENCES

- [1] Picard, Rosalind W. "Affective Computing." Cambridge, MA: Perceptual Computing Section, Media Laboratory, Massachusetts Institute of Technology, 1995.
- [2] Picard, Rosalind W. "Affective Computing for HCI." HCI (1). 1999.
- [3] Heller, Eva. "A Psicologia das Cores. ed." Espanha: Editora Garamond Ltda (2000).
- [4] Picard, Rosalind W. "Toward computers that recognize and respond to user emotion." IBM systems journal 39.3.4 (2000): 705-719.
- [5] Picard, Rosalind W., Elias Vyzas, and Jennifer Healey. "Toward machine emotional intelligence: Analysis of affective physiological state." IEEE transactions on pattern analysis and machine intelligence 23.10 (2001): 1175-1191.
- [6] Muller, Michael. "Multiple paradigms in affective computing." Interacting with Computers 16.4 (2004): 759-768.
- [7] Wang, Yi-Chi, and John M. Usher. "Application of reinforcement learning for agent-based production scheduling." Engineering Applications of Artificial Intelligence 18.1 (2005): 73-82.
- [8] El Kaliouby, Rana, Rosalind Picard, and Simon Baron-Cohen. "Affective computing and autism." Annals of the New York Academy of Sciences 1093.1 (2006): 228-248.
- [9] Elliott, Robert, et al. "Empathy." Psychotherapy 48.1 (2011): 43.
- [10] Kunzel, Michael Fernando, et al. "DIMI 3D-Companion Agent in a Learning Virtual Environment." 2011 Workshop and School of Agent Systems, their Environment and Applications. IEEE, 2011.

- [11] Liu, Chanjuan, Min Luo, and Chunhua Jiang. "Exploring the Operating Mode of Computational Stylistics: Taking Affective Computing as an Example." 2012 Fourth International Conference on Multimedia Information Networking and Security. IEEE, 2012.
- [12] Wu, Dongrui. "Fuzzy sets and systems in building closed-loop affective computing systems for human-computer interaction: Advances and new research directions." 2012 IEEE International Conference on Fuzzy Systems. IEEE, 2012.
- [13] Chittaro, Luca, and Riccardo Sioni. "Exploring eye-blink startle response as a physiological measure for affective computing." 2013 Humaine Association Conference on Affective Computing and Intelligent Interaction. IEEE, 2013.
- [14] Palmer, Stephen E., et al. "Music-color associations are mediated by emotion." *Proceedings of the National Academy of Sciences* 110.22 (2013): 8836-8841.
- [15] Ahn, Hyung-il, and Rosalind W. Picard. "Measuring affective-cognitive experience and predicting market success." *IEEE Transactions on Affective Computing* 5.2 (2014): 173-186.
- [16] Elliot, Andrew J., and Markus A. Maier. "Color psychology: Effects of perceiving color on psychological functioning in humans." *Annual review of psychology* 65 (2014): 95-120.
- [17] Han, Lingling, and Xiaodong Li. "The Appliance of Affective Computing in Man-Machine Dialogue: Assisting Therapy of Having Autism." 2014 Fourth International Conference on Communication Systems and Network Technologies. IEEE, 2014.
- [18] Novak, Domen, Aniket Nagle, and Robert Riener. "Linking recognition accuracy and user experience in an affective feedback loop." *IEEE Transactions on Affective Computing* 5.2 (2014): 168-172.
- [19] Adams, Andra, et al. "Decoupling facial expressions and head motions in complex emotions." 2015 International Conference on Affective Computing and Intelligent Interaction (ACII). IEEE, 2015.
- [20] Eid, Mohamad A., and Hussein Al Osman. "Affective haptics: Current research and future directions." *IEEE Access* 4 (2015): 26-40.
- [21] Faur, Caroline, et al. "A socio-cognitive approach to personality: Machine-learned game strategies as cues of regulatory focus." 2015 International Conference on Affective Computing and Intelligent Interaction (ACII). IEEE, 2015.
- [22] Gong, Rui, et al. "Investigation on factors to influence color emotion and color preference responses." *Optik* 136 (2017): 71-78.
- [23] Jin, ShengJu. "Social language affective computing using sparse representation." 2017 10th International Symposium on Computational Intelligence and Design (ISCID). Vol. 2. IEEE, 2017.
- [24] Zucco, Chiara, Barbara Calabrese, and Mario Cannataro. "Sentiment analysis and affective computing for depression monitoring." 2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM). IEEE, 2017.
- [25] Dong, Xiaoyang, et al. "Tree-Based Sentiment Dictionary for Affective Computing: A New Approach." 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS). IEEE, 2018.
- [26] Bakhtiyari, Kaveh, et al. "Ambiance Signal Processing: A Study on Collaborative Affective Computing." 2019 5th International Conference on Web Research (ICWR). IEEE, 2019.
- [27] Yang, Jinpeng, et al. "Research on Multimodal Affective Computing Oriented to Online Collaborative Learning." 2019 IEEE 19th International Conference on Advanced Learning Technologies (ICALT). Vol. 2161. IEEE, 2019.

# Analyzing the Credit-Assignment Problem in Deep Reinforcement Learning

José Aleixo Cruz 

Artificial Intelligence and Computer Science Lab (LIACC)

Faculty of Engineering of the University of Porto

Porto, Portugal

up201403526@fe.up.pt

**Abstract**—Deep reinforcement learning (DRL) is becoming a more relevant area of research, as it allows computational agents to learn complex tasks with evaluative feedback. One of the notable challenges of deep reinforcement learning is training in environments with sparse reward functions. A sparse reward signal aggravates the credit-assignment problem, which is the problem of determining the actions that lead to a certain outcome. In our work, we study the effect that the density of the reward function has on the learning process. We create two pairs of similar reinforcement learning environments, which differ only on the reward function and analyze the training results of different DRL methods. We conclude that even state-of-the-art methods are not able to cope with the credit assignment problem and reflect on the importance of reward function design.

**Index Terms**—deep reinforcement learning, credit-assignment problem, sparse reward, dense reward

## I. INTRODUCTION

Deep reinforcement learning (DRL) has become one of the most relevant and active areas of research in recent years. Researchers have successfully applied DRL techniques in adversarial games, such as Atari games [1], [2] and Go [3], [4], and in multi-agent games that require cooperation and competition, like DOTA 2 [5]. In some cases, agents trained with DRL surpass human performance at the task they learned.

One of the challenges for deep reinforcement learning is dealing with sparse rewards. Suppose that a learning agent has to control a robot’s arm and hand so that it grabs an object and puts it inside of a box. An example of a reward function can be giving the agent a positive reward when it reaches a state where the item is inside the box and a neutral (zero) reward for any other states. As the sequence of actions needed to receive any reward different than zero is too long, the agent may never reach the rewarding state; therefore, it may never learn the task. Sparse rewards also make it difficult to assign credit for the success or failure of a goal to each action taken, which is called the *credit-assignment problem* [6], [7].

Given the active research in deep reinforcement learning, there are plenty of algorithms and techniques that mitigate some of the problems which arise in a DRL context. With our work, we aim at presenting a broad and intuitive review of the most crucial reinforcement learning and DRL concepts to contribute to a better understanding of how DRL algorithms operate. With this knowledge, we intend to study the impact

that an algorithm’s specification has on its performance in a sparse reward environment and a dense reward environment.

The credit-assignment problem is one of the most significant obstacles in the current reinforcement learning landscape. By analyzing the performance of different algorithms, we hope to contribute with insights into this problem. These insights can not only help researchers choose a more appropriate algorithm for their studies, but also encourage the development of new solutions.

The rest of the paper is structured as follows. Section II introduces background information regarding reinforcement learning, deep reinforcement learning, and the credit-assignment problem. Section III presents a short review of works related to our own. Section IV displays the mathematical formalization of a reinforcement learning task. Section V contains information about the environments and tools used for experimentation. Section VI shows the results obtained along with the analysis of these results. Finally, Section VII contains the main conclusions of this work and considerations about future improvements.

## II. BACKGROUND

In this section, we present an overview of essential reinforcement learning and deep reinforcement learning concepts. Most definitions and descriptions are based on [8] and [9] and complemented with papers about the latest state of the art.

### A. Reinforcement Learning

In reinforcement learning, an agent interacts with the *environment* that surrounds it over time. At a given time step  $t$ , the agent perceives a set of environmental characteristics that make up a *state*  $s_t$  and selects an *action*  $a_t$  to take. Depending on the resulting state  $s_{t+1}$ , the agent gets a *reward*  $r_{t+1}$ . The objective of a reinforcement learning agent is to maximize the reward it receives over time.

An agent’s *policy* is a probability distribution that maps states to probabilities of selecting each available action. If an agent is following a policy  $\pi$  at time step  $t$ , then  $\pi(a | s)$  is the probability of  $a_t = a$  if  $s_t = s$ . The agent’s policy should enable it to choose a sequence of actions that leads to the best reward possible, which is its ultimate goal.

The *expected return* or *expected reward* of a state in a time step  $t$ , denoted  $G_t$ , is defined in (1), where  $\gamma$  is a



discount factor ( $\gamma \in [0, 1]$ ) that determines the importance of future rewards in updating current estimates, and  $T$  is the total number of time steps. The expected reward represents the cumulative sum of discounted rewards starting from instant  $t$ .

$$G_t \doteq \sum_{k=t+1}^T \gamma^{k-t-1} r_k \quad (1)$$

The *state-value function* (2), denoted  $v_\pi(s)$ , is the expected reward when starting in state  $s$  and following policy  $\pi$ , and the *state-action-value function* (3), denoted  $q_\pi(s, a)$ , is the expected reward of taking action  $a$  in state  $s$  and from that time forward following policy  $\pi$ .

$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t \mid s_t = s] \quad (2)$$

$$q_\pi(s, a) \doteq \mathbb{E}_\pi[G_t \mid s_t = s, a_t = a] \quad (3)$$

Initially, the agent does not know the value of any of these value functions, so it tries to estimate them by interacting with the environment and figuring out which actions lead to which states and rewards. One of the most common methods for estimating state-action-value, which is often called the *Q-value*, is doing the average of  $n$  observed rewards for a given action  $a$  in a state  $s$  (4).

$$Q(s, a)_n \doteq \frac{r_1 + r_2 + \dots + r_{n-1}}{n-1} \quad (4)$$

The memory requirements grow linearly with this method of value update. However, there is another way of calculating the average and keeping the memory requirements constant, described in (5). This update rule is the basis of **Temporal-Difference (TD)** reinforcement learning methods (10). The general form is described in (6).

$$Q_{n+1} \doteq \frac{1}{n} \sum_{i=1}^n r_i = Q_n + \frac{1}{n} [r_n - Q_n] \quad (5)$$

$$\begin{aligned} NewEstimate \leftarrow OldEstimate + \\ StepSize [Target - OldEstimate] \end{aligned} \quad (6)$$

The  $[Target - OldEstimate]$  expression is an *error* in the estimate. We can reduce this error by taking a step toward the *target*, which indicates the desired direction to progress. The main objective of a TD algorithm is to reduce the *error* over time until the current estimate converges. The *StepSize* parameter determines how much weight the newly found information has on the old value update. The impact of the *Target* value shows how important it is to define a suitable reward function for a problem.

One of the most popular TD methods is **Q-Learning** (11). Q-Learning estimates the state-action-value for each existing state-action pair. It is also a *tabular solution* reinforcement learning method because it stores the estimates of the value function in arrays or tables. We present the pseudo-code for

Q-Learning in Fig. 1. The update rule written on line 13 is a particular instance of (6).  $\alpha$  is the *learning rate*, which is analogous to *StepSize*, and  $\gamma$  is the *discount factor*.

```

1: for all  $s \in S$  do
2:   for all  $a \in A$  do
3:     Initialize  $Q(s, a)$  arbitrarily
4:   end for
5: end for
6: Set values for  $\alpha$  and  $\gamma$ 
7: for all episodes do
8:   Initialize  $s_0$ 
9:    $s_t \leftarrow s_0$ 
10:  repeat
11:    Choose  $a_t$  from  $s_t$  using policy derived from  $Q$ 
12:    Take action  $a_t$ , observe  $r_{t+1}, s_{t+1}$ 
13:     $Q(s_t, a_t) \leftarrow Q(s_t, a_t) +$ 
         $\alpha [r_{t+1} + \gamma \max_a [Q(s_{t+1}, a)] - Q(s_t, a_t)]$ 
14:     $s_t \leftarrow s_{t+1}$ 
15:  until  $s_t$  is a terminal state
16: end for

```

Fig. 1. Q-Learning algorithm pseudo-code

In problems where the state and action space are too large, storing Q-values for each possible state-action pair  $(s, a)$  is not feasible, due to hardware costs and limitations. *Approximate solution* methods are more suitable for this kind of problem. In these methods, the agent makes decisions on states it has never been in before based on previous encounters in similar states. As neural networks have revealed to be proper function approximators, researchers have used them to create *deep reinforcement learning* algorithms.

### B. Deep Reinforcement Learning

Training neural networks involves reducing the difference between the neural network's estimate for a value, given its input, and the actual value. This difference is the *error* or *loss*. To output more accurate values, the neural network must reduce its error through training. Training a neural network can be interpreted as the process of minimizing a certain *loss function*, that depends on the target values we wish to achieve.

*Value-based* deep reinforcement learning methods use *value networks*. A value network estimates  $v(s)$  or  $q(s, a)$ , that is, the network estimates the value of a state or a state-action pair. The **Deep Q-Network (DQN)** (12) method, for example, uses neural networks to approximate  $q(s, a)$ . It receives a state  $s$  as input and outputs the state-action value for each available action  $a$ . The neural network improves its estimates by adjusting its weights to minimize the current estimate error. If we take the update rule of Q-Learning, we verify that the error  $\epsilon$  of the estimate is (7). Therefore the loss function  $L$  the neural network should minimize is (8). When employing neural networks, it is important to make sure that samples are independent of each other and follow similar data distributions. To address this, instead of feeding the DQN with a mini-batch of sequential observations, such as the

last  $N$  observations, a mechanism called *experience replay* is used. To perform experience replay, we store a considerable number of observations in a buffer and, instead of feeding them sequentially to the neural network, we extract a mini-batch randomly. This reduces the correlation between observations, because we are using both recent as well as old observations, so they are more independent of each other and the agent doesn't "forget" what it had previously learned

$$\epsilon_t = r_{t+1} + \gamma \max_a [Q(s_{t+1}, a)] - Q(s_t, a_t) \quad (7)$$

$$L(t) = \epsilon_t^2 \quad (8)$$

*Policy-based* (also called *policy-gradient*) deep reinforcement learning methods use *policy networks*. In these methods, the policy  $\pi$  depends on a vector of parameters  $\theta$ , in contrast to value-based methods, where  $\pi$  depends on value functions  $v$  or  $q$ .  $\theta$  generally represents the weights of the policy network. One of the advantages of policy-based methods over value-based methods is that the policy may be a simpler function to approximate than the value function, particularly in large search spaces. Policy networks (Fig. 2) receive a state  $s$  as an input and output the policy itself, that is, the probability of choosing each available action, given the current parameters of the network ( $P(a | s, \theta)$ ).

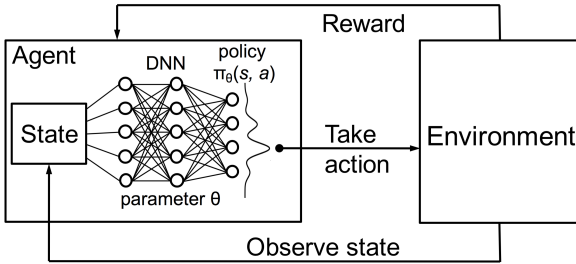


Fig. 2. Reinforcement learning with a policy network. [12]

Policy networks aim to improve the current policy  $\pi$ . To measure how good a policy with parameters  $\theta$  is, we use a performance measure  $J(\theta)$ . We can define  $J(\theta)$  as the expected reward of following the policy  $\pi$  with parameters  $\theta$  starting on initial state  $s_0$  [9]. As we want to maximize  $J(\theta)$ , the policy network's loss function will be  $-J(\theta)$ .

$$J(\theta) \doteq v_{\pi_\theta}(s_0) \doteq \sum_a \pi_\theta(a|s_0) q_{\pi_\theta}(s_0, a) \quad (9)$$

Neural networks often use gradient descent to minimize the loss function. Some methods explore the incorporation of using a baseline function to stabilize gradients and accelerate learning. *Actor-critic* methods are policy-based methods that also learn to approximate a value function. These are considered hybrid methods, as they combine approaches used in value-based methods and policy-based methods. The purpose of learning the value function is precisely to use it as a baseline during the learning process and assist on the policy update by

reducing variance. For example, let's say that for a given pair  $(s, a)$  the values of  $\pi_\theta(a|s)$  are  $[0.5, 0.3, 0.2]$  and the values of  $q_{\pi_\theta}(s, a)$  are  $[1000, 1001, 1002]$ . The variance of the product of the values is equal to  $Var(0.5 * 1000, 0.3 * 1001, 0.2 * 1002) = 232700$ . By using a simple constant baseline  $b(s) = 1000$ , we obtain  $q_{\pi_\theta}(s, a) - b(s) = [1, 2, 3]$ , so the variance then becomes  $Var(0.5 * 1, 0.3 * 2, 0.2 * 3) = 0.0433333$ , which is a much smaller value.

An *actor-critic* method has a learning model split in two components, depicted in Figure 3. One component, labeled the *actor*, determines the action to take on a given state. The actor is a reference to the learned policy, which dictates how the agent acts. The other component is called the *critic* and evaluates the impact of that action, which is a reference to the value function. *TD-error* (*temporal-difference error*) is the difference between the estimates of a value at different time steps, as we have previously introduced. The actor will update its values, and therefore the choices it makes, according to the TD-error calculated by the critic. The critic will also make adjustments to its values based on the TD-error.

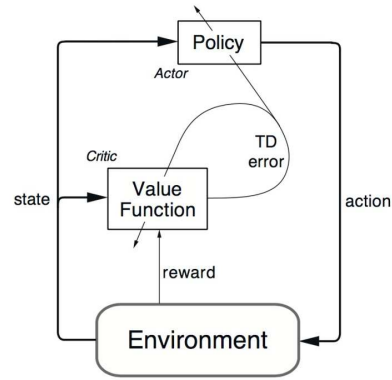


Fig. 3. The actor-critic method architecture. [13]

A common baseline to use when learning a policy is the advantage of taking an action  $a$  on a state  $s$ , given by  $A(s, a) = Q(s, a) - V(s)$ . The advantage function captures how good an action is compared to other actions that may be taken in the same state.

The **Advantage Actor-Critic (A2C)** [14] method combines both notions in its name. In A2C, the *critic* learns the advantage function, instead of the Q-values, thus reducing the high variance of policy networks and stabilizing the model.

Policy-gradient methods are generally too sensitive to the choice of the *StepSize* parameter from [6]. If the steps are too small, the learning progression is extremely slow. If the steps are too large, the model may overshoot an optimal region and end up with a noisy learning signal.

The **Trust Region Policy Optimization (TRPO)** [15] tries to fix this issue by setting a constraint [10] on how much a new policy can differ from the old one, using the Kullback–Leibler divergence (KL) [16], which measures the difference between two probability distributions.

$$\begin{aligned} & \underset{\theta}{\text{maximize}} \mathbb{E}_t \left[ \frac{\pi_{\theta}(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} A(s_t, a_t) \right] \\ & \text{subject to } \mathbb{E}_t [KL[\pi_{\theta_{old}}(\cdot|s_t), \pi_{\theta}(\cdot|s_t)]] \leq \delta \end{aligned} \quad (10)$$

With this constraint, the policy is kept in a trust region, so that it avoids taking larger steps than recommended.

An algorithm that improves on the concept of A2C and TRPO is the **Actor Critic using Kronecker-factored Trust Region (ACKTR)** [14]. ACKTR uses distributed Kronecker factorization [17] to optimize the calculation of the gradient update. This update can be 25% more expensive than A2C, but it leads to better results. It also uses a trust-region for updating policies.

The **Proximal Policy Optimization (PPO)** [18] improves on the TRPO trust-region concept by embedding the policy difference constraint on the objective function, instead of having a hard constraint. This facilitates and speeds up calculations while maintaining the policy in a trust region. Let  $r_t(\theta)$  denote the probability ratio. Then the objective function of PPO is given by [11], where  $\epsilon$  is a hyper-parameter.

$$\begin{aligned} & \underset{\theta}{\text{maximize}} \mathbb{E}_t [\min(r_t(\theta)A(s_t, a_t), \\ & \quad \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon)A(s_t, a_t))], \\ & \quad \text{where } r_t(\theta) = \frac{\pi_{\theta}(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \end{aligned} \quad (11)$$

The first term of the objective function is the objective function of TRPO. The second term clips the ratio between the old and new probability distributions of the policy, removing the incentive for changing the policy beyond  $[1 - \epsilon, 1 + \epsilon]$ . The minimum term is used so the final objective is a lower (pessimistic) bound of the unclipped objective.

### C. Credit-Assignment Problem

One of the first appearances of the credit-assignment problem (CAP) in literature was in the notes from Minsky [6]. In them, Minsky contemplated the difficulty of assigning credit to the individual actions of a complicated task. More generally, the CAP concerns determining how each component of a system has contributed to its overall performance. Minsky makes an intuitive description of the CAP: “Suppose that one million decisions are involved in a complex task (such as winning a chess game). Could we assign to each decision element one-millionth of the credit for the completed task?” [6, p. 20].

Depending on the system’s architecture and task, we can distinguish different types of credit-assignment. In a multi-agent system, for example, we may identify the *structural* credit-assignment problem, which means we do not know what components of the system are responsible for the achieved outcome. In single-agent domains with complex tasks, we may identify the *temporal* credit-assignment problem, which concerns how an action taken at a particular time affects the outcome. Our work focuses only on the performance of the

current state of the art reinforcement learning methods when faced with the **temporal CAP**.

## III. RELATED WORK

In 1984, Sutton did a detailed analysis of the performance of reinforcement learning algorithms regarding the temporal credit-assignment problem [7]. Sutton tested the algorithms in *nonassociative* and *associative* tasks. Nonassociative tasks are tasks where the agent does not receive any stimulus (or observation) and must try to choose the action that leads to the best reward. There is no mapping between stimulus and output; hence it is nonassociative. Associative tasks are tasks in which the agent must form a mapping from input (stimulus) to output, to execute the most rewarding action. One of the products of Sutton’s study was the introduction of methods with the actor-critic architecture introduced in Section III.

More recently, researchers have tried to reduce the effects of the temporal credit-assignment problem using deep learning methods. Liu et al. [19] developed an approach that decomposes an episodic task’s reward to each time step in the trajectory. It takes a sparse reward and tries to learn a dense surrogate reward function that approximates the temporal credit-assignment of the episodic reward. The dense reward should provide more information than the sparse reward, speeding up learning. Results showed that the learned reward signal improved learning and sample efficiency in locomotive control tasks. Harutyunyan et al. [20] propose a new family of reinforcement learning methods, where the value function is written using hindsight rather than foresight. Instead of attempting to figure out how a particular action affects future outcomes (foresight), the researchers try to understand, given an outcome, how relevant were the past decisions (hindsight). While its scalability was not tested, this approach addresses some of the critical issues in credit assignment.

Regarding deep reinforcement learning benchmarking, [21] tested several state of the art algorithms in continuous control tasks, concluding that while some algorithms had a satisfactory performance in some cases, they mostly performed poorly in hierarchical tasks, which involve learning low-level skills that the agent can use in high-level decisions. Temporal credit-assignment in hierarchical tasks is hard to achieve. When agents only receive the reward at the end of the episode, the delay between the reward signal and the actions is too significant. Nagendra et al. [22] analyzed the performance of reinforcement learning methods applied to the traditional cart-pole problem, where the goal of the agent is to balance a pendulum that is pivoted in a cart. The researchers observed that value function approximation had superior performance than policy gradient actor-critic, which in turn had better performance than Q-Learning.

While many recent works focus on developing algorithms that try to improve on the temporal CAP problem, we were not able to find a paper that follows the vein of Sutton’s work and compares performance regarding the temporal CAP in different tasks between the current state-of-the-art algorithms.

We hope to contribute to filling that gap with the results of this work.

#### IV. PROBLEM FORMALIZATION

Firstly, we define a reinforcement learning task as a **Markov Decision Process (MDP)**, where  $S$  is a finite set of states,  $A$  is a finite set of actions,  $P_a(s_t, s_{t+1})$  is the probability that an action  $a$  on state  $s_t$  at a given time  $t$  will lead to state  $s_{t+1}$  at a time  $t + 1$ , and  $R_a(s_t, s_{t+1})$  is the immediate reward  $r_{t+1}$  of transitioning from state  $s_t$  to state  $s_{t+1}$  after taking action  $a$ . A MDP must satisfy the *Markov property*, which determines that future states may only depend on the current state, not on the sequence of states that preceded it. In mathematical terms, this translates to  $P(s_{t+1} | s_t) = P(s_{t+1} | s_1, s_2, \dots, s_t)$ .  $P_a(s_t, s_{t+1})$  is the *transition function*, which represents the environment’s dynamics. An *episode* is the period of time between the initial state and the terminal state. A *trajectory*  $T$  is the sequence of states visited, actions taken and rewards received during an episode ( $T = (s_0, a_0, r_1, s_1, a_1, r_2, \dots, s_{t-1}, a_{t-1}, r_t, s_t)$ ).

For the current problem, we will make the following assumptions:

- We will assume that our agent does not know the transition function; hence we will use *model-free* methods, where the agent must interact with the environment to learn.
- We will assume that the environment is stochastic, that is,  $0 < P_a(s_t, s_{t+1}) < 1$ ; therefore the agent is never sure about what state it will transition into.

We will also make the following definitions, which should only be considered in the context of this work:

- We define a **sparse reward function** as a reward function that has a zero value for every transition, except for a transition that leads to the goal state, which provides a reward  $r$  (12).
- We define a **dense reward function** as a reward function that has a non-zero value for at least one transition in every state (13).

$$R_{a_{\text{sparse}}}(s_t, s_{t+1}) = \begin{cases} r \neq 0, & \text{if } s_{t+1} \text{ is the goal state} \\ 0, & \text{otherwise} \end{cases} \quad (12)$$

$$\exists a \in A : R_{a_{\text{dense}}}(s_t, s_{t+1}) \neq 0 \quad (13)$$

We wish to observe how the cumulative reward that our agent receives in an episode (14) develops during its training stage in two types of environment: environments with a sparse reward function and environments with a dense reward function. We will compare the rewards obtained with different reinforcement learning methods.

$$R(T) = \sum_{r \in \mathcal{R}} r, \text{ where } \mathcal{R} \text{ is the set of rewards in } T \quad (14)$$

#### V. METHODS AND MATERIAL

To test the performance of different reinforcement learning algorithms in regards to the CAP, we built two pairs of simple reinforcement learning environments<sup>1</sup> in Python using the OpenAI Gym [23] interface.

We called one pair of environments `cpa`, as in **correct-parity analyzer**. In these environments, the agent’s observation is a single integer number  $n \in [0, 9]$ . The set of possible actions is  $A = \{0, 1, 2\}$  where  $a = 0$  means the agent believes the number it observes is even and  $a = 1$  means it believes it is odd.  $a = 2$  is an action that does not serve any purpose, but slightly increases the size of the action space. The agent needs to guess correctly the parity of  $x$  observations, given  $3x$  attempts. In one environment, called `cpa_dense`, it gets a positive reward  $r$  for each correct parity guess and 0 otherwise. In the other, called `cpa_sparse`, it gets a positive reward  $xr$  only when it guesses  $x$  correct parities and 0 in every other state. For our experiments, we set the values  $r = 1$  and  $x = 100$ .

We based the other pair of problems on the mountain car task, which [24] first described. We adapted the already existing implementation of the `MountainCarContinuous-v0` environment from the OpenAI Gym repository<sup>2</sup>. In the mountain car task, the agent controls a car in a one-dimensional track, positioned between two hills. The objective is to drive up the hill to the right of the car; however, the car’s engine is not strong enough to climb the hill in a single pass. Therefore, the only way to succeed is to drive back and forth to build up momentum. In this environment, the agent’s observation is a pair of rational numbers  $(c_p, c_v)$ ,  $c_p \in [-1.2, 0.6]$ ,  $c_v \in [-0.07, 0.07]$ , which describe the car’s position  $c_p$  and velocity  $c_v$ . The agent may choose any action between the interval  $[-1.0, 1.0]$ . An action  $a$  with a value  $-1.0 \leq a < 0$  pushes the car to the left, while an action with value  $0 < a \leq 1.0$  pushes the car to the right. In one environment, called `mc_sparse`, it receives a reward  $r = 100$  once it reaches the top of the hill at position  $g_p = 0.5$  and a zero-reward in every other state. In the other, called `mc_dense`, it receives a reward equal to the difference between the current car’s position and the objective position in every state.

We chose these particular tasks as they are simple enough to run multiple experiments on an average personal computer, and we can observe the influence of the sparsity of the reward function in the credit assignment problem.

As we implemented the environment with the Gym interface, we can use reinforcement learning algorithms that already exist and are compatible with Gym. In our experiments, we use the Stable-Baselines [25] algorithms, namely the DQN, A2C, ACKTR, and PPO algorithms. These particular algorithms use Tensorflow [26] to build the artificial neural networks that are used as policy and value networks. Using these Stable-Baselines, we hope that implementation details do not have any effect on performance comparison, as the

<sup>1</sup>Available at: <https://github.com/jazzchipc/feup-mic>

<sup>2</sup>Available at: <https://github.com/openai/gym>

algorithms use the same tools and belong to the same authors. The discount factor and learning rate parameters used for each algorithm is described in Table I. These parameters are the default parameters used by Stable-Baselines and have not been specifically tuned.

TABLE I  
PARAMETERS FOR EACH OF THE USED METHODS

Description	ACKTR	PPO	A2C	DQN
Discount factor	$\gamma = 0.99$	$\gamma = 0.99$	$\gamma = 0.99$	$\gamma = 0.99$
Learning rate	$\alpha = 0.25$	$\alpha = 0.0025$	$\alpha = 0.007$	$\alpha = 0.005$

We trained different agents with each algorithm in the *cpa* environments. Because all methods are probabilistic, to get more accurate results, we ran training 10 times for each algorithm for 100000 steps. In the *mc* environments we didn't train DQN agents, as the action space is continuous, and using *max* in a continuous function is not simple. We only ran each algorithm once for the *mc* environments. The running computer had an Intel i5-6300HQ CPU and 8GB of RAM. We have not used any GPU to accelerate processing.

For each training process, we extracted information at the end of each episode and saved it. Such information includes the total reward received at the end of the episode, the number of timesteps of the episode, and the time elapsed.

## VI. RESULTS AND ANALYSIS

### A. *cpa* environments

From the gathered results, we were able to associate the episode reward with the number of training timesteps taken until the end of that episode, constructing a time series. Different runs of the same algorithm had different timesteps, so to average the episode reward of the runs, we interpolated the respective time series and calculated the mean for each 5 timesteps interval. We present the average reward per episode of each algorithm in the *cpa\_dense* and *cpa\_sparse* environments in Figures 4 and 5, respectively. We clipped the number of timesteps in Figure 4 to exclude irrelevant information and used a moving average with a window of 10 timesteps to smooth the curves. We also present the average time per step for each algorithm in Table II and the number of episodes without any reward in the *cpa\_sparse* environment in Table III.

TABLE II  
AVERAGE TIME PER STEP (IN MS) IN THE *CPA* ENVIRONMENTS

Environment	ACKTR	PPO	A2C	DQN
<i>cpa_dense</i>	0.9262	0.6119	0.7881	2.2806
<i>cpa_sparse</i>	0.9402	0.6118	0.7908	2.2846

In our experiments, we have set the end of training as a step limit, which influences how we should interpret the achieved results. The algorithm specifications mentioned in Section III affect the time of each learning step. For example, PPO aims at lowering the computational cost for each step, while DQN

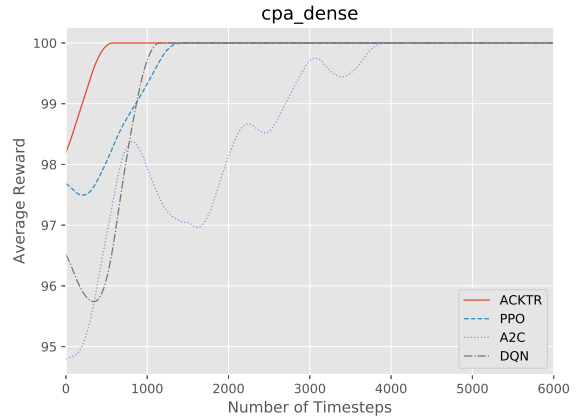


Fig. 4. Average episode reward per number of training timesteps in the *cpa\_dense* environment.

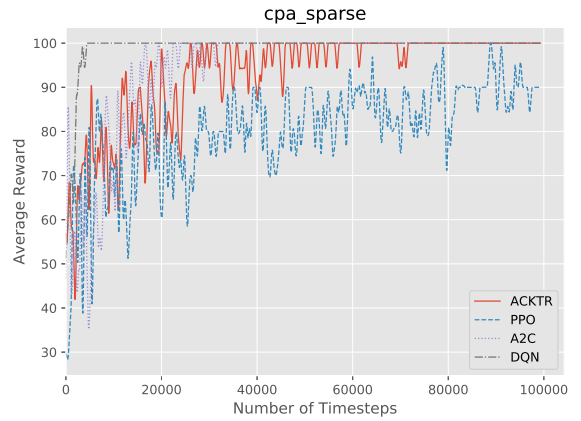


Fig. 5. Average episode reward per number of training timesteps in the *cpa\_sparse* environment.

processes much more information in a single step. Also, as we can observe, the average time for each PPO step is the lowest of all algorithms, ACKTR is a bit more expensive than A2C, and DQN is the most expensive. While using a time limit for training was possible, we refrained from doing this because we were unsure if other processes that were running in the same machine limited CPU or RAM availability.

Another relevant implementation detail is that in each environment, once the agent reaches the target number of correct parity guesses, the episode ends, which means that agents that

TABLE III  
NUMBER OF EPISODES WITH NO REWARD IN THE *CPA\_SPARSE* ENVIRONMENT

	ACKTR	PPO	A2C	DQN
Episodes without reward	237	666	193	31
Total number of episodes	4217	4018	4304	8944
% episodes without reward	5.62%	16.57%	4.48%	0.35%

guess correctly in fewer steps can do more episodes. This detail is essential for interpreting accurately the results we display in Table III.

The most striking observation we can make out of the results is that the algorithms were **much** faster to learn the task in the environment with the dense reward function than in the environment with the sparse reward function. Therefore, all algorithms were better at assigning credit to actions. We expected this behavior, as the dense reward function provides instant feedback about the value of the action taken in a given timestep. The sparse reward, on the contrary, produces a delayed reward signal, that is only available at the end of the episode.

Another observation is that PPO was not able to learn the optimal policy every time it ran on the `cpa_sparse` environment, while we expected it to be among the fastest algorithms. We attribute this result, in part, to the starting point of the PPO policy. As all of these methods involve some randomness in deciding what actions to take, being in a more favorable starting point helps getting to the global optimum quicker. PPO had the worst starting point of all algorithms. Therefore, it took longer to find a policy that led to success. In fact, it was the algorithm with higher percentage of episodes without any reward. Without any feedback, there were no hints to improve the policy.

DQN was the algorithm that learned with fewer steps in the sparse reward environment, which can be explained by the experience replay buffer it uses. Both environments have small action and observation spaces. As DQN saves previous interactions and reuses them in training, it achieved a higher sample-efficiency than the remaining policy-gradient algorithms; that is, it learned the optimal policy in fewer steps. Also, rewards do not depend on future states, but solely on the action chosen; therefore, value-based methods may be at an advantage in comparison to policy-based methods. As it learned quicker, it was also able to run more episodes than the other algorithms, having the lowest percentage of episodes without any reward. However, the average DQN step took more than double the time than the other algorithms, which makes it the slowest processing algorithm. We would expect that in a more complex environment, DQN’s performance would not be so good, as large search spaces favor policy-based algorithms.

### B. *mc* environments

For the mountain car environments, we present a different perspective on the data. In Fig. 6 and Fig. 7 we present the number of episodes completed by the agents trained in the `mc_dense` and `mc_sparse` environment, respectively. We intend to analyze whether the task was successfully completed or not, hence we ignore the value of the reward and just take into account the number of finished episodes. In Table IV we display the average time for each step in milliseconds. We have not used the DQN algorithm in this environment, because DQN is designed for discrete action spaces.

In these environments, an episode only ends once the agent achieves the goal position, contrary to the `cpa` environments where it would end even when the agent had not reached the objective. We can verify that in the `mc` environment, some of the agents do a lot better with the sparse reward function than with the dense reward function. This behavior is the opposite of what we verified with the `cpa` environments. We believe this is due to the quality of the dense reward we designed for this problem, which exposes that creating denser reward functions may not always lead to the outcome we expect when designing them.

The results of each algorithm also support what we have hypothesized about the starting point profoundly affecting the rest of the learning process. Once an agent reaches the objective state, it learns to get there again faster. However, the longer it takes to reach the final state, the longer will be the learning process.

The time per step of the algorithms in this environment was higher than in the previous environments, probably due to the more complex action and observation spaces. The relative performance remained the same, with PPO being the fastest and ACKTR being the slowest.

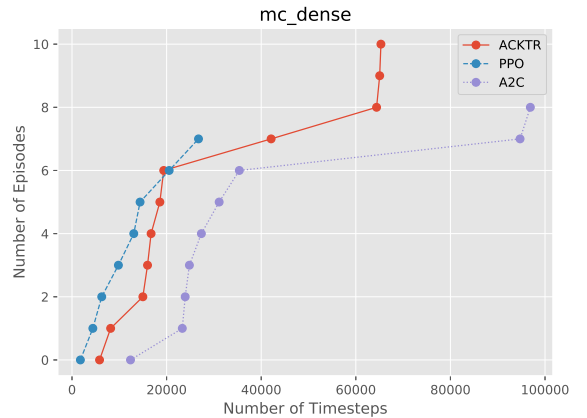


Fig. 6. Number of episodes completed per number of training timesteps in the `mc_dense` environment.

TABLE IV  
AVERAGE TIME PER STEP (IN MS) IN THE `mc` ENVIRONMENTS

Environment	ACKTR	PPO	A2C
<code>mc_dense</code>	1.2071	0.7679	0.9261
<code>mc_sparse</code>	1.1304	0.7344	0.8911

We verify that the temporal credit-assignment problem has a significant influence on the performance of DRL algorithms. Denser reward functions lead to faster learning. However, it is harder to design a proper dense reward function than a sparse reward function, either because the designer must acquire domain knowledge or because there is not enough information to calculate the reward. Also, creating denser reward functions might inadvertently “steer” the learning agent to a particular

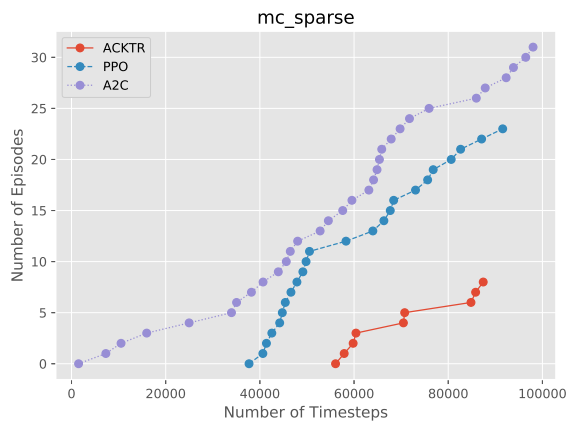


Fig. 7. Number of episodes completed per number of training timesteps in the `mc_sparse` environment.

behavior, narrowing its field of exploration. As sparse rewards are preferred, the temporal CAP remains one of the notable obstacles of current DRL methods.

## VII. CONCLUSIONS

We have tested implementations of the ACKTR, PPO, A2C, and DQN deep reinforcement learning algorithms in two pairs of similar environments, which only differed in the density of the reward function. We observed that the temporal credit-assignment problem has a significant impact on the speed of learning. In one environment, all algorithms performed worse in the environment with the sparse reward function than in the one with the dense reward function. In the other, the opposite occurred. We have also observed that the starting point of a policy impacts the rest of the learning process. The delayed reward signal of sparse reward functions makes it harder to correctly attribute credit to actions, given the outcome. An agent may never learn an optimal policy if the reward function is too sparse or has a bad design, even with significant training. These results show that while the reinforcement learning algorithm is essential for training, designing a suitable reward function for a task is critical to having success.

In our work, we have only two simple tasks to compare performances in dense and sparse reward spaces. Possible future work could involve using a multitude of tasks, with different types of action and observation spaces, and different reward functions. Understanding how algorithms react to different circumstances can provide researchers with better information for their works.

## REFERENCES

- [1] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.
- [2] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

- [3] D. Silver, A. Huang, C. J. Maddison *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, p. 484, 2016.
- [4] D. Silver, J. Schrittwieser *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, p. 354, 2017.
- [5] OpenAI, "Openai five," <https://blog.openai.com/openai-five/>, 2018.
- [6] M. Minsky, "Steps Toward Artificial Intelligence," pp. 8–30, 1961.
- [7] R. S. Sutton, "Temporal credit assignment in reinforcement learning," Ph.D. dissertation, Graduate School of the University of Massachusetts, 1984. [Online]. Available: <https://elibrary.ru/item.asp?id=7414440http://www.incompleteideas.net/papers/Sutton-PhD-thesis.pdf>
- [8] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, 2nd ed. MIT press, 2018.
- [9] Y. Li, "Deep reinforcement learning: An overview," 2017.
- [10] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Machine learning*, vol. 3, no. 1, pp. 9–44, 1988.
- [11] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3–4, pp. 279–292, 1992.
- [12] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*. ACM, 2016, pp. 50–56.
- [13] C. Yoon, "Understanding actor critic methods and a2c," <https://towardsdatascience.com/understanding-actor-critic-methods-931b97b6df3f>, 2019.
- [14] Y. Wu, E. Mansimov, S. Liao, R. B. Grosse, and J. Ba, "Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation," *CoRR*, vol. abs/1708.05144, 2017. [Online]. Available: <http://arxiv.org/abs/1708.05144>
- [15] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, "Trust region policy optimization," *CoRR*, vol. abs/1502.05477, 2015. [Online]. Available: <http://arxiv.org/abs/1502.05477>
- [16] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.
- [17] J. Martens and R. B. Grosse, "Optimizing neural networks with kronecker-factored approximate curvature," *CoRR*, vol. abs/1503.05671, 2015. [Online]. Available: <http://arxiv.org/abs/1503.05671>
- [18] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, vol. abs/1707.06347, 2017. [Online]. Available: <http://arxiv.org/abs/1707.06347>
- [19] Y. Liu, Y. Luo, Y. Zhong, X. Chen, Q. Liu, and J. Peng, "Sequence modeling of temporal credit assignment for episodic reinforcement learning," *CoRR*, vol. abs/1905.13420, 2019. [Online]. Available: <http://arxiv.org/abs/1905.13420>
- [20] A. Harutyunyan, W. Dabney, T. Mesnard *et al.*, "Hindsight credit assignment," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle *et al.*, Eds. Curran Associates, Inc., 2019, pp. 12467–12476. [Online]. Available: <http://papers.nips.cc/paper/9413-hindsight-credit-assignment.pdf>
- [21] Y. Duan, X. Chen, R. Houthoofd, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," *CoRR*, vol. abs/1604.06778, 2016. [Online]. Available: <http://arxiv.org/abs/1604.06778>
- [22] S. Nagendra, N. Podila, R. Ugarakhod, and K. George, "Comparison of reinforcement learning algorithms applied to the cart-pole problem," in *2017 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2017*, vol. 2017-Janua, 2017, pp. 26–32.
- [23] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," 2016.
- [24] A. W. Moore, "Efficient memory-based learning for robot control," Ph.D. dissertation, University of Cambridge, 1990.
- [25] A. Hill, A. Raffin, M. Ernestus, A. Gleave *et al.*, "Stable baselines," <https://github.com/hill-a/stable-baselines>, 2018.
- [26] M. Abadi, A. Agarwal, P. Barham *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: <http://tensorflow.org/>

# SESSION 3

## Robotics and Simulation

Teaching a Robot Precision Ball Sports using Deep Reinforcement Learning

*Liliana Antão*

Realistic wheelchair simulator using gazebo and ROS

*Ana Beatriz Cruz, Armando Sousa, and Luís Paulo Reis*



# Teaching a Robot to play Boccia and Bocce using Deep Reinforcement Learning

Liliana Antão

Faculty of Engineering, University of Porto  
Rua Dr. Roberto Frias, 4200-465 Porto, Portugal  
Email: up201502833@fe.up.pt

**Abstract**—In the last years, there has been considerable interest in making robots learn to perform typically human activities and behaviors. One of the most common abilities searched is learning play sports. This has been made possible with the increase of automated learning, promoted by higher computational power and also, reinforcement learning (RL) algorithms.

This paper implements a beginner robotic player for precision ball sports like *boccia* and *bocce*. A new simulated environment is created a seven degree-of-freedom (DoF) robotic arm is able to learn how to throw different types of balls towards the goal (the jack) using deep reinforcement learning. The simulator was adapted from OpenAI Gym environments and uses the MuJoCo realistic physics engine. A brief comparison of the convergence of different RL algorithms is performed. Several ball weights and various types of materials correspondent to *bocce* and *boccia* are tested, as well as different friction coefficients. Results show that the robot achieves a maximum success rate of 92.7% and a mean of 75.7% for the best case with a lighter ball, and 85.2% max and 69.4% mean for the heavier while playing a single-player version of these sports with the DDPG+HER algorithm.

# Realistic wheelchair simulator using gazebo and ROS

Ana Beatriz Cruz

*Artificial Intelligence and Computer Science Laboratory (LIACC)  
and Faculty of Engineering, University of Porto (FEUP)*  
Porto, Portugal  
up201403564@fe.up.pt

Armando Sousa

*INESC TEC - INESC Technology and Science and  
Faculty of Engineering, University of Porto (FEUP)*  
Porto, Portugal  
asousa@fe.up.pt

Luís Paulo Reis

*Artificial Intelligence and Computer Science Laboratory (LIACC)  
and Faculty of Engineering, University of Porto (FEUP)*  
Porto, Portugal  
lpreis@fe.up.pt

**Abstract**—The evolution of intelligent wheelchairs with new systems to control them and help the user to be more independent has been remarkable in recent years. Since these systems have a big impact on the life quality in people with disabilities, it is crucial that it is suited for the final user. This study proposes a 3D motorised wheelchair model with robotic tools to be used in simulation environments and helps the development and validation of new approaches. This model uses Robotic Operating System (ROS) tools to help the addition of sensors and actuators. With the ROS-Nodes it is easy to add new features and controllers with robotic ideals. The Gazebo framework was used to create the simulation environments. It is also possible to easily change the model of the wheelchair by the real one, and apply the features tested in the simulation. It was possible to control the model with keyboard arrows, in different world and simulated scenarios. Real world and simulated wheelchair are tested and their behaviour is similar. It is hoped that such similarity will boost the development speed of further developments.

**Index Terms**—Intelligent wheelchair, Robotics, Gazebo, ROS, Controller

## I. INTRODUCTION

Many people have locomotion problems and cognitive limitations. They are dependent on other people to perform essential actions in their daily lives, such as turn on the television. Generally, these people have motorised wheelchairs to help them move around. Since wheelchairs are a necessary support for people with disabilities, and they use them in the most significant part of the day, why not equip wheelchairs with systems that allow the users to be more independent?

Before getting a final model of a robotic system, it is necessary to perform tests. Not always everything goes according to plan and testing in real systems could mean loss of money when the systems fail. Because of that, simulators that enable to add, remove or modify robotic systems in wheelchairs easily helps their study and improvement. Besides that, tests in the real world could place people in danger in the case of model failures. With realistic simulators, it is possible to test the models and find problems before testing them in the real world.

This work presents a new wheelchair model simulator and a basic model of a 6-wheeled motorised wheelchair with central traction for simulation system based on ROS and Gazebo. Based on a real wheelchair, it has all its physics features to make the model as realistic as possible. The model in gazebo allows to add new sensors and actuators and validate the behaviour of the wheelchair. Figure 1 shows the real motorised wheelchair used for this project.



Fig. 1. Motorised wheelchair used to replicate in simulated environment.

This realistic model allows to add new robotic systems in the wheelchair and make several realistic simulations with physics constraints. It is also possible to test and validate the model in different type of situations in a simulation environment. The identification of failures or the necessity of other components is possible with this model without losing money and endangering people.

In sum, the goals of this project are developing wheelchair 3D model in a simulated environment with behaviour as closely as possible with the real wheelchair in different scenarios of navigation. This features delivers the realistic model and allows to make tests more reliable in a simulated environment.

The model should be developed based on the real wheelchair presented in Figure 1.

The remaining of the paper is structured as follows. Section II presents some wheelchair simulators developed in the last years, presenting their important goals and conclusions for this work. The following section, III, consists of detailing the methodology, with the characterisation of the developed model and the final system architecture. Section IV specifies the tests made to validate the model, with the discussion of the respective results. In the end, in Section V, the major conclusions about the work developed are presented.

## II. LITERATURE REVIEW

To improve the development of new models of wheelchairs and test specific systems or situations, some authors have worked on the development of simulators.

With the evolution of intelligent wheelchairs and with the increasing number of human-computer interfaces, new ways to help the user have emerged, with its limitations, to control the wheelchair as autonomous as possible. So, the human and wheelchair interaction have been the main topics studied. It is crucial to understand if the users can control an intelligent wheelchair before they get one. Other simulators enable the users to train the control of the wheelchair, before using it in the real world.

A survey made by [1] in 2014 presents eighteen wheelchairs intelligent simulators. The simulators mentioned above are focused on improving intelligent wheelchair driving, either by the user, or the intelligence of the wheelchair.

Most recently [2] created the *IntellSim* to help the users train its navigation in different simulated environments, by controlling the manual wheelchair or with a joystick in the electric wheelchair cases. The authors concluded that the *IntellSim* can be used for both patient training or evaluation, design and development of semi-autonomous intelligent wheelchairs. The author [3] follows a similar approach with a hand motion controller and, in spite of the successful results given by the users of the developed simulator, the author considers the system with reduced kinematic performance in virtual reality than the real world.

Serious games for individuals with disabilities or motor disorders, like Boccia, have been used to help people with disabilities to develop and rehabilitate their cognitive capabilities. A realistic simulator for Boccia game helps the users to practice in their houses [4].

All mentioned simulators have the main focus on human wheelchair interface and the user control. However, all systems have poor sensors. The study of [5] uses UNITY to create a virtual environment to control a wheelchair and try to add some real sensors. However, this simulator still has few sensors, and the investment on them could be lost if they are not suitable or in unsuccessful tests.

Some robotics studies tried to create systems to test sensors and actuators in simulations environments before assembling one in the real world. The study of [6] presents a simulation toolkit to help the development of robotic systems, more

specifically with grasping. The developed system focuses on helping the development and testing of new algorithms for robotics, modelling of the environments, modelling of actuators, sensors and contacts.

Training intelligent models, finding issues at an early stage of developments and testing the usability of some sensors and actuators are fundamental steps for the development of intelligent systems. The case of the intelligent wheelchairs, these studies are essential to get a final system and make it suitable for its user that has disabilities.

In conclusion, the main focus of recent studies with wheelchair simulators is the user experience and the improvement of its navigation. However, the kinematic performance of these systems is still weak. Some developed robotic simulators have kinematic constraints, but not specific to wheelchairs.

## III. MODEL

ROS is a widely used robotics distributed framework, equipped with libraries and tools to help create robot applications [7]. The number of contributions with open source tools is vast and in constant evolution. There is already a significant quantity of real sensors available to test in ROS. The Gazebo framework allows to visualise and test the realistic 3D models with the dynamics of the environment that a robot could encounter [8]. It is also compatible with ROS, so it was chosen to develop the realistic simulator of wheelchairs [9].

For this work, it was necessary the development of a new wheelchair model, compatible with Gazebo and very similar to the real model of this study. All features of the developed model are explained in Section III-A

### A. Development of the model

There are many tools for developing 3D models. However, not all the frameworks have physics constraints, and the final models are not compatible with ROS and Gazebo. Blender is a free, open-source framework that allows to design 3D models with built-in support for many computer-aided design (CAD) file formats and rigid body kinematics [6].

However, Gazebo needs to import robot models in *.sdf* format and Blender initially does not export that. To get around this problem, the Phobos<sup>1</sup> add-on was added in Blender, which is an open-source tool that helps to develop and edit the robot models, and export them in a format compatible with Gazebo.

Three parts are fundamental to create robot models to make it as most realistic as possible. They are links, joints and plugins for sensors and actuators. The link parts are composed by the visual elements, collision geometries and inertial features. The Figure 2 resumes the structure of a *.sdf* file. Each part is explained in the next Sections.

Figure 3 presents final model of the wheelchair obtained in this project.

<sup>1</sup><https://github.com/dfki-ric/phobos/wiki>

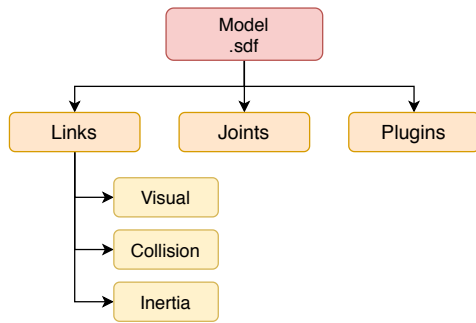


Fig. 2. Structure of a .sdf file.



Fig. 3. Final model of the wheelchair.

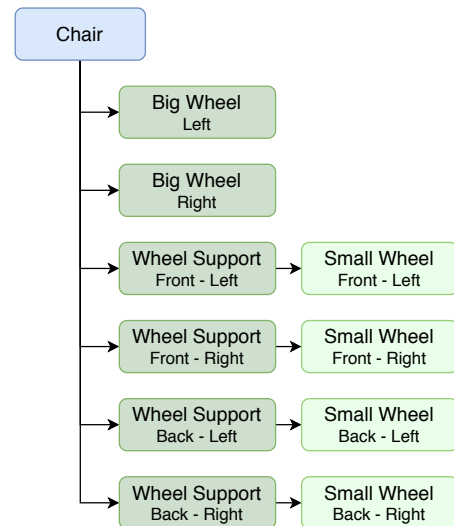


Fig. 4. Skeleton of all links of the model.



Fig. 5. Visual component of the chair in wheelchair model.

1) *Links*: In .sdf models, the skeleton is composed of links. Links are like bones in the human skeleton, and they are organised in a parent-child relationship.

The wheelchair model developed in this work is composed of eleven links. The main one is the chair with two armrests and the footrest. This part is the root of the rest of the model. Four wheels supports and two bigger wheels are children of the main link. In the end, all wheels support links have each child, that is the smaller wheel.

Each link has its specific features and located in a specific place relative to the root link. As mentioned before, each link is composed of visual, collision and inertial parts.

a) *Visual Elements*: The visual element specifies the shape used by the rendering engine. Sometimes, the visual part could have complex shapes (not representable with simple geometric shapes, like cubes), so it is necessary to create meshes for all visual parts and import them in a .sdf file. For the case of the wheelchair model developed, meshes were used for all visual parts with .dae files, to bring the model the most similar as possible with the real wheelchair.

The visual part of the wheelchair model is composed of three main parts. The first one is the chair with two armrests and the footrest, presented in Figure 5. This visual element is called just one time by the *chair link*.

A cylinder that supports the smaller wheels is the other visual part of the wheelchair, as shown in Figure 6. The visual

part is used four times, that represents the four supports for the smaller wheels. All cylinders have the same shape, just with different orientation and position.

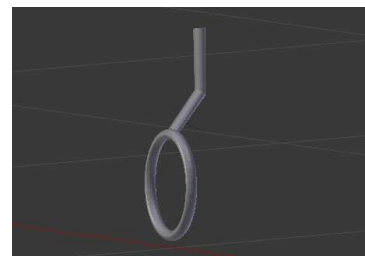


Fig. 6. Visual component of the smalls wheels support in wheelchair model.

The last visual component is the wheel presented in Figure 7. This element was developed to seems like a real wheel. Since the real wheelchair used in the base of this project has six wheels, the developed mesh is called six times by the six

links that correspond to each wheel.

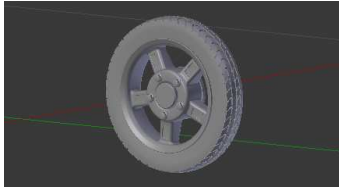


Fig. 7. Visual component of the wheel in wheelchair model.

The two central wheels are larger than the other four. For the two links that correspond to those, the visual part is increased by a proportional factor equivalent to the difference of the wheels in the real wheelchair model.

The wheels of the real wheelchair have 0.35 meters and 0.19 meters of diameter in the biggest and smaller wheels respectively. The proportion of the simulated wheels should be  $0.35/0.19 = 1.84$ , so, the size of the biggest wheels are 1.84 bigger than the smaller wheels. With this, it is possible to guarantee that the velocity of the real wheelchair is the same as the simulated model.

*b) Collisions:* A collision element encapsulates a geometry that is used for collision checking. When the models are simple, the collision and the visual elements are the same. However, in the case that the models are complex, it is essential to simplify the collision element to help improve performance. It is considered simple geometries elements like cubes and cylinders. Since the wheelchair model developed in this project has complex visual elements with meshes, the collision parts were simplified.

It was created some collisions elements adequate for each link. The developed collision parts guarantee that all possible collision zones (the limits of the wheelchair) have a corresponding collision element, to make a simulation as close as possible to the real wheelchair.

The *chair link* is the biggest one, and it has some concave and convex shapes. The fundamental limits of the wheelchair for a possible collision during its navigation was identified. Figure 8 shows with red lines the identified essential limits for the collision elements.

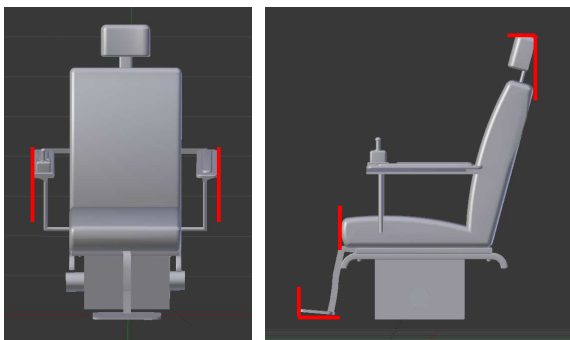


Fig. 8. Limits for collision zones of the wheelchair model.

Three boxes compose the collision elements of the *chair link*. Two boxes, a) and b) in Figure 9, protects all limits in the chair. The c) box of the Figure 9 has the same dimensions as the footrest.

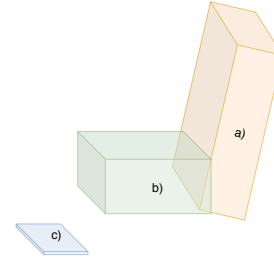


Fig. 9. Collision boxes of *chair link*.

The only parts that are outside of those collisions and are fundamental for a realistic simulation are the six wheels. For each wheel, a simple cylinder with the same dimensions of the visual part was defined. The cylinders keep the rotation property of the wheels.

Figure 10 shows the model with all collision elements in their respective positions.



Fig. 10. Final model of the wheelchair and collision boxes.

*c) Inertia Information:* An accurate simulation requires physically plausible inertial parameters. They are the mass, centre of mass location, and the moment of inertia matrix of all links. All links were considered with homogeneous density.

The mass of each link was considered the same as the real wheelchair. The table I presents the approximate mass of each part of the real wheelchair that has a corresponding link in the wheelchair model.

Since it was considered the homogeneous density, the centre of mass of each link is located in their respective geometric centroid.

*MeshLab* was used to obtain the moment of inertia matrix for all links. This program can calculate the moment of inertia of a mesh considering its respective mass.

TABLE I  
MASS OF EACH PART OF THE REAL WHEELCHAIR

Link	Mass
Chair	145 Kg
Cylinder	0.5 Kg
Small Wheel	1 Kg
Big Wheel	3 Kg

Figure 11 shows the centre of inertia considered for inertia in its respective link.

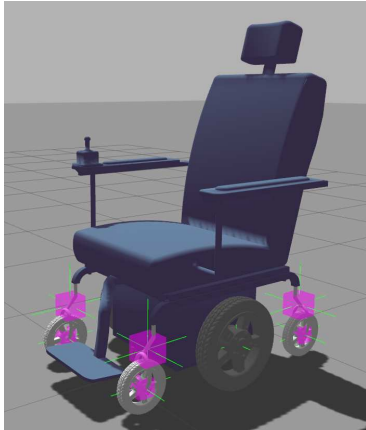


Fig. 11. Inertia elements in final wheelchair model.

2) *Joints*: As mentioned before, a *.sdf* model has a skeleton with bones, that are the links with parent-child relationships. With this organisation, it is possible to define the connections between two links and how a link moves relative to its parent link with the use of joints. Figure 12<sup>2</sup> shows the relation between links (green) with a joint (blue).

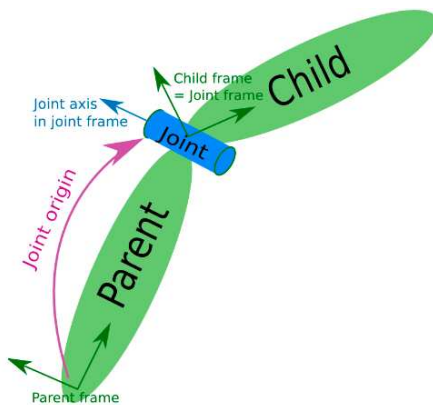


Fig. 12. Structure of two links and one joint between them.

Joints define the relationship between parent and child links, by choosing the type of the joint and some other specific features for each type, such as the axis of rotation, and joint limits.

<sup>2</sup><http://wiki.ros.org/urdf/XML/joint>

Figure 13 shows the relation of all links with their respective joints in the developed wheelchair model.

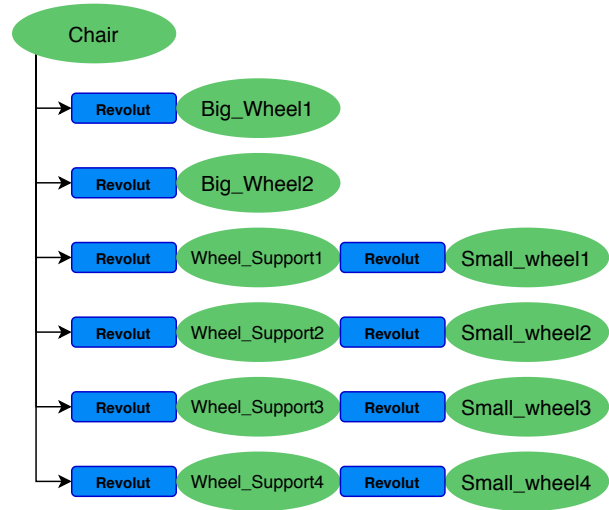


Fig. 13. Hierarchic structure of the links and joints in the wheelchair model.

As it is possible to see, all joints have the same type, revolute, which is a hinge joint that rotates along the axis and has a limited range specified by the upper and lower limits. Since the cylinders and wheels are free to rotate along an axis, revolute is the most suited type of joint. The difference between cylinders and wheels are the axis on which they rotate. All joints are presented in Figure 14.

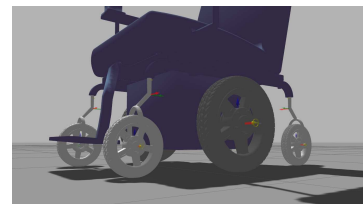


Fig. 14. Joints presented in the wheelchair model.

3) *Plugin*: It is necessary to add the elements that make the model move and some way to communicate with them, to control the wheelchair model in gazebo with ROS tools.

The real wheelchair moves with two motors by differential drive, one in each of the centre wheels. By controlling these two motors, it is possible to move the wheelchair with a linear and angular velocity. The control of the wheelchair could be made by different devices like joystick or head sensors.

There is a ROS plugin called *libgazebo\_ros\_diff\_drive* that was used to connect the model with ROS by using a library called *libgazebo\_ros\_diff\_drive.so*. This plugin has some parameters that are necessary to define in order to move the model correctly. Identify the two joints that correspond to the wheels that make the robot move. Other important features are the subscribed topic that (*/cmd\_vel*) that contains the linear and the angular velocity that robot should move, and

the update rate of the values. For this case, the update rate is 50 messages per second.

### B. System Architecture

The use of ROS helps the integration of different devices to control the wheelchair in a simulated or real environment. The communication between nodes implemented in ROS is made through topics using the publisher-subscriber topology [10].

A simple architecture was created to test the wheelchair model on gazebo, with a simple controller that reads the keyboard arrows and translate them in linear and angular velocity. Figure 15 shows the architecture of the system.

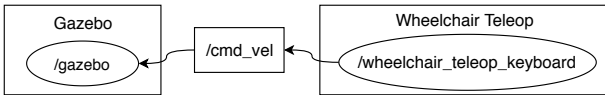


Fig. 15. Box diagram with system architecture.

## IV. TESTS AND RESULTS

The developed wheelchair model was imported to Gazebo to test it in a simulated environment. With *roslaunch* tool, the gazebo simulation begins with a wheelchair in an empty world, for the first validations. Two tests for the model validation were made. The first test validates the communication between the different ROS-nodes and the movement of the wheelchair in Gazebo. The second test verifies the behaviour of the wheelchair in different scenarios, comparing its behaviour with the real wheelchair.

a) *Testing commands and communication:* As mentioned before, to control the wheelchair model in a simulated environment, it was created a simple control with the keyboard arrows. These commands were processed in two different forms, the continuous increase of the linear and angular velocity and the other with maximum velocity when one arrow is pressed or zero if no arrow selected. For both strategies, the wheelchair responds correctly to the linear and angular velocity. In the case of the continuous increase of velocities, the wheelchair stops increasing the velocity when it reaches the maximum velocity established to the model.

The topic that has the velocity is */cmd\_vel*. As mentioned before, the node */wheelchair\_teleop\_keyboard* publishes to this topic and is subscribed by the wheelchair model included in ROS-node */gazebo*. Figure 16 shows it is working correctly and Figure 17 shows the existent messages in real-time in that topic.

```

bea@bea-rog:~/wheelchair_ws$ rostopic info /cmd_vel
Type: geometry_msgs/Twist

Publishers:
 * /wheelchair_teleop_keyboard (http://bea-rog:38289/)

Subscribers:
 * /gazebo (http://bea-rog:39031/)
  
```

Fig. 16. Information about the topic */cmd\_vel*.

```

angular:
  x: 0.0
  y: 0.0
  z: 0.0
---
linear:
  x: 0.3
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.0
---
linear:
  
```

Fig. 17. Example of messages in topic */cmd\_vel*.

To validate the versatility and the ability to change the control of the wheelchair, a node that communicates with an implemented multimodal interface for wheelchair by sockets was developed. Receiving commands for that existing tool, it was possible to control the simulated wheelchair as a real one.

b) *Testing model behaviour:* The physics constraints were tested for simple navigation conditions.

The wheelchair model has four wheels to keep the wheelchair balanced, two in front and other two in the back of the wheelchair. These wheels follow the wheelchair on its way to keep it balanced.

The same commands were sent to test the behaviour of the simulated wheelchair model compared with the real wheelchair. Next figures show the position of the four small wheels in different scenarios in the simulated and real model. First they go forward, Figure 18, next they go backward, Figure 19, and in the last one, Figure 20, turn around.

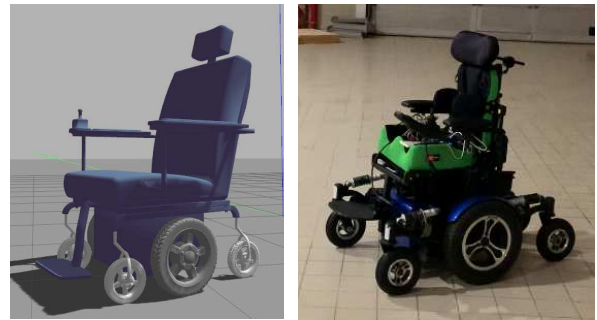


Fig. 18. Movement of the real and simulated wheelchair going forward.

In all these figures, it is possible to see that the four wheels can follow the movement of the wheelchair in a simulated environment like the real wheelchair in the same situation. With this control, it was possible to see that the simulated wheelchair has the same behaviour as the real wheelchair when it receives the same message controls.

Figure 20 presents the wheelchairs when they turn around. The four small wheels get a circular position in Gazebo like in the real wheelchair.

Another feature tested in the simulated wheelchair was the collision with rigid objects. Some boxes were added in gazebo world. Intentionally, the wheelchair went in the object



Fig. 19. Movement of the real and simulated wheelchair going backward.



Fig. 20. Movement of the real and simulated wheelchair turning around.

directions, and the moment of the collision, the wheelchair stopped. Figure 21 shows an example of collision with a box.

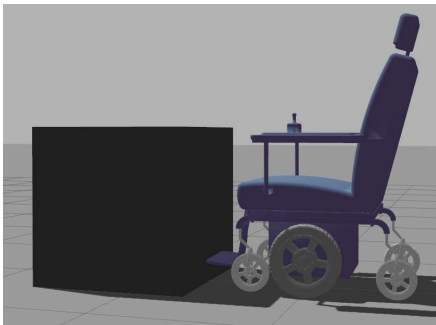


Fig. 21. Collision test between simulated wheelchair and a box.

## V. CONCLUSION

The work developed proposes a motorised wheelchair model used in realistic simulations. This model uses ROS to help the implementation of the robotics ideas and create an architecture based in ROS-nodes that can easily change for other nodes and topics. The use of a ROS environment allows the simulation of real sensors in any position of the wheelchair and test it in diverse simulation environments. The developed wheelchair model can be used to add more sensors, create new controllers and test them in a simulation environment before implementing it in the real world with the final user.

While state of the art shows the focus of the wheelchair simulation with user experience and other robotic simulations with kinematic constraints, but not for wheelchairs, the proposed model merge the two approaches. It is expected that the kinematic behaviour of the simulated wheelchair can be very similar to the real world wheelchair.

The presented model is minimalist in sensors and actuators, similar to the real world wheelchair, because of the need for its evolution. The real model of the wheelchair has an elevator that could place the wheelchair user at a higher height. This feature was not considered in this model. If this is regarded as an essential feature, it should be taken into account in future versions.

In future versions, engineering parts and tools could be added in the developed model. With this simulator, it is possible to test new electronic devices and controls for intelligent motorised wheelchairs, making its development more efficient and with the lower costs.

## REFERENCES

- [1] B. M. Faria, L. P. Reis, and N. Lau, "A survey on intelligent wheelchair prototypes and simulators," in *Advances in Intelligent Systems and Computing*, vol. 275 AISC, no. VOLUME 1. Springer Verlag, 2014, pp. 545–557.
- [2] B. M. Faria, L. P. Reis, N. Lau, A. P. Moreira, M. Petry, and L. M. Ferreira, "Intelligent wheelchair driving: Bridging the gap between virtual and real intelligent wheelchairs," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9273. Springer Verlag, 2015, pp. 445–456.
- [3] G. Tao and P. S. Archambault, "Powered wheelchair simulator development: implementing combined navigation-reaching tasks with a 3D hand motion controller," *Journal of NeuroEngineering and Rehabilitation*, vol. 13, no. 1, p. 3, dec 2016. [Online]. Available: <http://www.jneuroengrehab.com/content/13/1/3>
- [4] B. M. Faria, J. D. Ribeiro, A. P. Moreira, and L. P. Reis, "Boccia game simulator: Serious game adapted for people with disabilities," *Expert Systems*, vol. 36, no. 3, pp. 1–11, 2019.
- [5] R. Chotikunanan, M. Sangworasil, P. Chotikunanan, T. Matsuura, and N. Thongpance, "Electric wheelchair simulation from unity 3D for controller test," *BMEiCON 2017 - 10th Biomedical Engineering International Conference*, vol. 2017-Janua, pp. 1–4, 2017.
- [6] B. León, S. Ulbrich, R. Diankov, G. Puche, M. Przybylski, A. Morales, T. Asfour, S. Moio, J. Bohg, J. Kuffner, and R. Dillmann, "Open-GRASP: A toolkit for robot grasping simulation," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6472 LNAI, pp. 109–120, 2010.
- [7] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.
- [8] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, pp. 2149–2154, 2004.
- [9] W. Qian, Z. Xia, J. Xiong, Y. Gan, Y. Guo, S. Weng, H. Deng, Y. Hu, and J. Zhang, "Manipulation task simulation using ROS and Gazebo," *2014 IEEE International Conference on Robotics and Biomimetics, IEEE ROBIO 2014*, pp. 2594–2598, 2014.
- [10] V. Costa, R. Rossetti, and A. Sousa, "Simulator for teaching robotics, ROS and autonomous driving in a competitive mindset," *International Journal of Technology and Human Interaction*, vol. 13, no. 4, pp. 19–32, 2017.



# SESSION 4

## Intelligent Systems II

MSDS-OPP: Operator Procedures Prediction in Material Safety Data Sheets

*Eliseu Pereira*

Offensive assessment in social networks

*Carlos Tomás*

Finding Emotions in Screenplays for Genre Classification

*Inês Nunes Teixeira*

# MSDS-OPP: Operator Procedures Prediction in Material Safety Data Sheets

Eliseu Pereira

*SYSTEC - Research Center for Systems and Technologies*

*Faculty of Engineering, University of Porto*

Porto, Portugal

eliseu@fe.up.pt

**Abstract**—The digitalization and tracking of products across the supply chain is one of the challenges of Industry4.0. The collected historical data contains crucial information about their constitution and safety constrains. Material Safety Data Sheets (MSDS) are the main source of information of safety issues about products. However, they have incomplete information, which causes operator confusion, that could be critical in some situations. The usage of multi-label text classification and Natural Language Processing (NLP), would make easier the information retrieval and allow the prediction of the material procedures, given the other MSDS fields. For that, it was built the Material Safety Data Sheet Operator Procedures Prediction Tool (MSDS-OPP), which scrapes MSDS from the web, extracts topics from the files, processes the textual data and using multi-label text classification predicts which will be the operator procedures. That automated task approach allows a more simple search of MSDS's and the extraction of crucial information, which improves the operator knowledge about the products handled in the factory, giving him a more summarized report. The results prove that both text processing pipelines (e.g., TF-IDF transformer) and text classifier (e.g., K-Nearest Neighbors and Decision Tree) perform well with high values of precision, recall and F1-score. In general, the tool has their limitations, but for operator recommendations would be a good solution, giving some traces about the steps to perform.

**Index Terms**—Natural Language Processing, Material Safety Data Sheet, Text Classification

## I. INTRODUCTION

Industry4.0 is emerging as a new industrial paradigm, allowing the manufacture of customized products (smart manufacturing), the monitorization of the industrial process, or the shop floor resources optimization [1]. The actual trend is to digitalize the shop floor equipment, enabling the understanding of the industrial process. The data collection process focuses on associate to each product the corresponding data allowing the product tracking and the visualization of their historical data. The historical data enables the identification of futures product defects and the estimation of patterns according to that defects. To complement these data usually is added some information about safety issues and danger materials present in the composition of the final product. These materials have restrictions and dangers to the operator usage and handing; typically, this information has the format of a text document.

A Material Safety Data Sheet (MSDS) [2] is the standard text document that describes in a detailed way the elaborated ingredients, the health hazards, the emergency procedures, and

other fields. This document has a conventional structure, with a clear description, to be easily readable by the operators in case of an emergency. These fields are organized into categories such as chemical data, health hazards, personal operator protection, and operator procedures. Usually, the MSDS's have as main representant the chemical products due to the MSDS structure; however, in the other industries, the usage of a standard document, like the MSDS, would help to understand the dangers of a particular product.

The main limitation for the usage of MSDS as a primary source of information from a material is the variance between documents written by different companies. This variance difficult the document understanding and, in extreme cases, allows the MSDS structure mutation according to each company. A solution to address this restriction is an approach based on Natural Language Processing (NLP) that automatically extracts the knowledge from an MSDS and generates the procedures to the operator take in case of an emergency.

An approach based in NLP and Machine Learning (ML), would allow the usage of incomplete information (e.g., the MSDS only has the ingredients data or the health hazards), which is common due to the different material manufactures and concern on share information in the MSDS data [3]. Additionally, these processed data could be annexed to the historical product data, allowing the users/operators to access these data independently of their role in the supply chain (e.g., operator, transporter or consumer). In certain situations, it's mandatory to have a quick response by the operator, performing a set of actions to neutralize that danger. The description of the material procedures needs to be clear about the steps to complete and always available. Additionally, the search engine must have a quick response time to avoid any delay in the operator actions.

The development of the Material Safety Data Sheet Operator Procedures Predictor (MSDS-OPP) intends to accomplish those goals. This tool scrapes the MSDS from the web, extracts each section from the MSDS file, processes the text, predicts the several procedures (fire fighting, material storage, or material disposal), and helps summarizing the MSDS content. More concretely uses multi-label text classification to predict the procedures, associating one classifier to each one (fire fighting, storage, and disposal). The classifier input features are extracted from the chemical data, health hazards, and personal

protection categories, using NLP algorithms. These algorithms filter the tokens and n-grams from the text and convert that textual data to numerical arrays to feed the classifier. The classifier labels are transformed from the procedures text description (also present in the MSDS file) to binary class labels, using a parallel pipeline different from the one used in the features.

The main contributions of the MSDS-OPP are 1) the scrapping of MSDS files from the web, due to the difficulty of obtaining that data (not very common or in some cases paid); 2) the extraction of knowledge from the data sheets, that could help the operator searching for MSDS fields; and 3) the operator procedures prediction, which allows a quick prediction of the steps to perform in situation of fire, material storage or material disposal.

Given that, the paper structure starts with a brief description of the *Literature Review* in section II, where first, the author explains some background concepts, like the MSDS structure. The second part of the *Literature Review* section is the related work, which describes similar projects done in the area of the MSDS's and projects that use multi-label text classification and Natural Language Processing techniques. The next section is the *Methodologies* (section III), which describes the applied algorithms and models, always comparing their capabilities and how to fit in the problem. The evaluation of the implemented methods occurs in the *Experiments & Results* (section IV), where different performance metrics allow the comparison between approaches. The last section is the *Conclusions & Future Work* (section V), which contains the discussion of the results, the applications of the implementations, the tool advantages and limitations, and the future work approaches to improve the project.

## II. LITERATURE REVIEW

The MSDS's are a source of information about dangerous products in several sectors like the medical, the industrial, or the agricultural. An automatic approach to analyze that data would benefit these sectors. However, before start deploying models and NLP pipelines, it's useful to understand the problem and the data itself. The data analysis must focus on the MSDS structure, validating which sections are significant for the problem and which procedures have the potential to be useful for the operator needs. After that, the analysis must focus on similar works in the area regarding the gaps existing in the literature.

### A. Background

Across the literature [3] [4] exist a discussion between what an MSDS should contain. The typical structure includes eight different sections: 1) Chemical Identity and Source Data, 2) Hazardous Ingredients, 3) Physical and Chemical Characteristics, 4) Fire and Explosion Hazard Data, 5) Reactivity Data, 6) Health Hazard Data, 7) Precautions for Safe Handling and Use, and 8) Control Measures. Table I presents a more detailed view of the data that provides from each section. There each

row contains a subsection, with their data type (e.g., textual, numerical, boolean) and their category.

To easily understand the content of an MSDS, Table I contains a reorganized structure, divided into four different categories of 1) chemical data, 2) health hazards, 3) personal protection, and 4) operator procedures. The chemical data includes information about the ingredient constitution of the material; and also includes other chemical properties like the boiling point, the vapor density and pressure, the flash point, or the evaporation rate. Additionally, it incorporates a short descriptions of the material appearance and decomposition products. The health hazards category includes the carcinogenicity specifications, a description of the effects of overexposure, and a short explanation of other health hazards. The personal protection data defines the equipment needed to handle the material; in this case, it specifies the gloves, eyes, and respiratory equipment to use. It also contains information about the ventilation in the area and some additional precautions to take in care. The last category is the operator procedures; here, it's specified the protocol to take into account in some situations. These situations could be emergencies like fire or some other casual actions like material storage or disposal. Due to emergencies, it's essential to have a tool that quickly given the material returns the emergency procedures to make by the operator. Additionally, as shown in Table I, it's also specified the manufacturing company and the product name as general information data.

TABLE I  
MSDS STRUCTURE

Subsection	Data Type	Grouping Category
product name	text	general info
company name	text	general info
ingredients	text	chemical data
decomposition products	text	chemical data
materials to avoid	text	chemical data
vapor pressure	numeric	chemical data
vapor density	numeric	chemical data
boiling point	numeric	chemical data
flash point	numeric	chemical data
evaporation rate	numeric	chemical data
material appearance	text	health hazards
carcinogenicity	text	health hazards
effects overexposure	text	health hazards
first aids	text	health hazards
eyes protection	text	personal protection
respiratory protection	text	personal protection
protective gloves	text	personal protection
ventilation	text	personal protection
protective equipment	text	personal protection
fire fighting	text	operator procedures
storage precautions	text	operator procedures
disposal procedures	text	operator procedures
spill procedures	text	operator procedures

Although the current structure has its limitations, such as their ambiguity, that often causes delays in diagnosing the occupational diseases, which places a further risk to the operator to develop long-term disorders [3]. Another constraint is the omission of vital information regarding the protected formula as a trade secret. In particular cases, the information

is underestimated, such as the flash point [5], where one-third of the products have a lower flash point value than the specified in the MSDS. The underestimation of the flammability of the product can cause severe problems, like ignitions. Furthermore, the little information about diseases associated with the product represents a lack for material usage. Regarding that, the MSDS is a useful starting point for handling some emergencies such as chemical contamination [4].

### B. Related Work

As related work, the focus was to identify the typical approaches used in the MSDS analysis. Most of the works related to MSDS focus on their structure and limitations [3] [4] [5] [6], alternatively other publications emphasize the search engines and databases used to manage the MSDS [7] [8]. In general, there are a few publications that use an NLP or ML approaches to analyze an MSDS dataset; the most similar ones focus on the drug side effects analysis [9] [10] [11] and text classification in general [12] [13] [14].

The MSDS availability is a concern, usually, the traditional organizations/repositories managing that data are the Registry of Toxic Effects of Chemical Substances (RTECS) [15], the Hazardous Chemicals Data Base (HSDB) or the Occupational Safety & Health Administration (OSHA) [8]. With the growth of the Internet, some search engines like *Google* or *Yahoo* [16] allow quick access to the MSDS data. Nevertheless, these data could be irrelevant, incomplete, or even false, which makes them not 100% trustful [7]. To avoid these situations have emerged some specialized sites/repositories that provide reliable and free data, like the *msdsonline.com* or the *hazard.com*.

In the universe of applications of NLP and ML methods, one of them is the drug indications and side effects prediction. The PISTON [9] allows the prediction of drug indications and side effects using topic modeling and Natural Language Processing (NLP). This tool extracts relations between drugs, genes, and diseases from literature data, finding sentences when both terms co-occur (topic) and compute their probability. The resulting probability values construct a drug-topic probability matrix. On top of the matrix is built a classifier that predicts the indications and side effects of drugs. Also, in this sector, NLP methods are used to automatically map drug product labels in the Medical Dictionary for Regulatory Activities (MedDRA) [10].

NLP tools are the core of decision support systems [11] that assist the human decision providing useful information about the problem. In both industrial and medical environments, it's crucial to make decisions with all the knowledge about the issue. Mainly, in medical situations is essential to identify the right risk factors, new treatments or medications, and associate concepts with certain diseases [11], for that a tool that predicts an initial guess will accelerate the process of diagnostic and treatment of diseases. Translating to the industrial applications, some situations like emergencies (e.g., fire) or daily work tasks (e.g., material storage), would benefit from the usage of these tools [14].

Text classification allows the label of documents according to several categories, it could be used to label federal reports,

movie reviews, or cooking recipes. The process typically starts with a dataset of labeled documents; these documents provide features that feed the classifier to predict which label fits well. There are variants for the steps used in the document classification pipeline. The classifier could use weight terms according to the term relevance [12]. The representation of the features could mutate depending on the implementation; in some situations, it is preferable to use the typical bag-of-words representation; other representations are the words embedding or the graph embedding [12] [13]. The larger area of classification problems branches into different fields, in particular 1) the one-class classification, where the model output is only one class, 2) the multiclass classification, which has multiple classes as model output but only one activated, and 3) the multi-label classification, which has multiple classes with multiple activation's simultaneously. Regarding the multi-label text classification, there are different approaches in the literature, e.g., the usage of neural networks [17]. Similar, to the automated text processing, in the multi-label text classification field there are few publications related to MSDS's. The authors focus on other areas, such as the medical, which targets to the multi-label text classification of medical documents according to their diseases [18] or the usage of multiples labels (e.g., cough and fever) in the clinical data description of the patient health, to improve the disease diagnostic [19].

In general, these NLP works don't focus on the MSDS analysis, and the publications based in the MSDS center on their structure and limitations. Most of the time, they use a human hand analysis, which delays the process. Also, these approaches are human dependent, which difficult the adaptability of new operators to the shop floor when they need to handle these products. One of the alternatives to accelerate the analysis process is the usage of NLP methods for the knowledge extraction and ML techniques to the operator procedures prediction. This additional tool would help both operators or physicians, given them supplementary information about the material. Their usage would increase the infrastructure/factory productivity, and add further details about the product safety issues, to the material historical data.

### III. METHODOLOGIES

The MSDS-OPP allows the prediction of the operator procedures for disposal, storage, and fire situations. To accomplish these goals, the MSDS-OPP performs most of the steps of an end-to-end NLP project. The tool architecture, in Figure 1, starts with the data acquisition from the web, then using text processing techniques extracts the features and class labels, with them, trains one classifier to each procedure, and finally reconstructs the classifier output to a procedure description. In more detail the MSDS-OPP architecture has five different stages: 1) the data scrapping, that gets the MSDS files from the web; 2) the feature extraction, which extracts features from the MSDS files and applies NLP methods to feed the classifiers with clean data; 3) the label transformation, that converts the label procedure text description in a set of class labels, which are used as targets to train and validate the classification

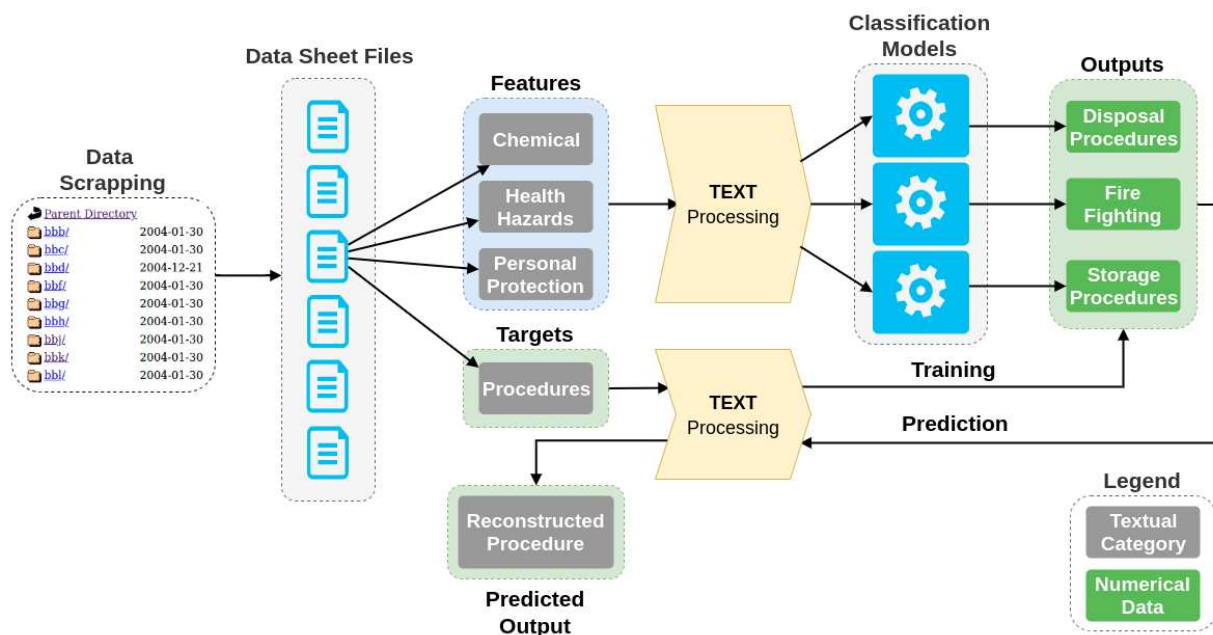


Fig. 1. MSDS-OPP Architecture

model; 4) the classification models, which implement and validate different classification models to predict the operator procedures; and 5) the description reconstruction, that given the predicted class labels transforms these tuples in a standard description of the procedures. Both classification models and text processing pipelines use the scikit-learn [20], also the pipelines use the NLTK [21] framework, which enables a simpler usage of NLP techniques.

#### A. Data Scrapping

MSDS files are hard to find on the web; in most cases, the user needs to pay for the dataset. Because of that, the MSDS-OPP scrapes the dataset from the site *hazard.com*. There are stored many different types of data sheet files; the MSDS-OPP uses the *F2* data sheet type, which is well structured and easy to scrape. The repository stores the files grouped in HTML "folders", which are composed of links each one associated with one MSDS file. So each HTML page contains a set of links parsed using the *beautiful soup* [22] framework, which makes easier the link parsing. Using the MSDS file link, it's downloaded the page from the repository and converted from HTML to a text file. This process searches iteratively all the links present in each folder.

The scraped data contains 236500 data sheet files, which corresponds to 1.3GB; the folder structure is the same present in the repository. That's huge among of data, which allows an appropriate training and validation of the classification models and the NLP methods.

#### B. Feature Extraction

The feature extraction part is one of the stages where are applied the NLP techniques. These techniques allow a

generic and robust extraction of features from the original text. To efficiently implement these methods is used a pipeline structure, where each algorithm is a step in the pipeline, and the data flows through all the steps.

Before applying the NLP pipeline, the MSDS sections described in Table I are extracted from each file using regular expressions. To accelerate the extraction is adopted a multi-processing approach, dividing the MSDS files by *N* workers. The mined sections converge to a table, where each row corresponds to one document. Then the parts associated with the chemical data, health hazards, and personal protection categories are transformed into input features for the classifiers.

The pipeline implemented to transform an input text in a set of numerical features to feed the classifier applies a sequential process using different methods in each step, Figure 2. First, the characters pass to lower case, and it's used a tokenizer to transform the plain text in a collection of word tokens. Next, a filter removes the stop words and the punctuation from the terms collection. From them are built one-grams and bi-grams based on the term frequency. Then the tokens collection is vectorized into a matrix with the tokens vocabulary as columns, and each row containing the number of occurrences of a token in the respective document.

The next step in the pipeline is the transformation of the number of occurrences of each token in the *term frequency-inverse document frequency* (TF-IDF) matrix. This method allows the understanding of how vital is a term in a particular document. The term frequency (TF) component calculates the fraction between the number of times that a token appears in a document and the total number of terms in the document. The inverse document frequency (IDF) calculates the ratio

between the total number of documents and the number of them containing the token. The TF-IDF is the product of both term frequency and inverse document frequency, where the IDF is constant to each term, and TF is variable between documents. This statistical method allows a term-weighting approach that reflects the importance of a term in a document. Also, the method allows the elimination of common terms (i.e., connectors) that doesn't bring to much value for the classifier.

The result of the entire pipeline is a matrix that has the terms vocabulary as columns and the term weight associated with each document as rows. From that matrix are selected the 150 more relevant terms, which feed the classification model, like features. The selection criteria choose the features (tokens) that have a higher weight in the TF-IDF matrix, which enables a more clean and determinant data to feed the classifier. So, when tested the features extraction component, if the method finds a token present in the document, it activates the corresponding feature as the input of the classifier.

### C. Label Transformation

The label transformation uses a different approach due to the limited number of class labels. Here the MSDS dataset provides a text description as a procedure, which isn't a binary label understandable by the classifier. For that, the procedure description text mutates into a set of labels that the classifier uses as targets. The transformation pipeline contains methods like Name Entity Recognition (NER), Position Tagging (POS-TAG), or Syntactic Dependency (DEP).

The label transformation, as the feature extraction, consumes text data obtained by regular expressions, in specific the operator procedures sections. Due to the usage of a classifier per procedure, it's mandatory to have a label transformation pipeline per each one. Mainly because each classifier has different labels, according to the procedure (e.g., the label "use air supplied rescue equipment" for the fire fighting procedures classifier and the label "incinerate absorbed material" for the disposal procedures classifier).

The label transformation pipeline, Figure 2, starts with the tokenization of the text description. For each sentence are extracted their entities and noun-chunks (noun plus words describing the noun). The extraction of noun-chunks allows a more ambiguous search for the key terms of the sentence. The repeated terms are filtered between the entities and the noun-chunks, resulting in a new set of tokens. Based on the extracted terms, the sentence is retokenized, the resulting tokens are tagged using position tagging, and it's analyzed the syntactic dependencies. Then for each entity and noun-chunk is checked their dependency tag; if it's an attribute or a direct object (*dobj*), it's extracted the subject from the sentence and added to the relation, with the entity or noun-chunk. If the dependency tag is an object of a preposition (*pobj*), the extracted relationship is the entity/noun-chunk plus the previous prepositional modifier.

Instead of using a TF-IDF transformer, like on the features extraction component, the label transformation pipeline uses the output of the vectorization step as class labels for the

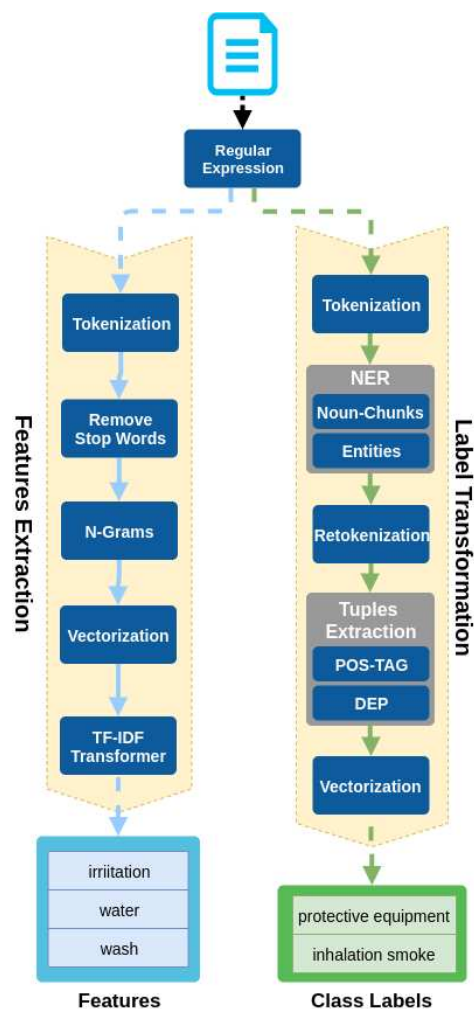


Fig. 2. Feature Extraction and Label Transformation - NLP Pipelines

classifier output. As a result, the pipelines produce a set of relations, from them, are selected the 20 more frequently, according to their distribution in the training set (Figure 4). Those relations will be the class labels of the classifier, used for the classifier training, and their predictions validation.

### D. Classification Models

After extracting both features and class labels from the MSDS file, it's applied a text classifier to predict the MSDS operator procedures based on the input features (chemical data, health hazards, and personal operator protection). The current structure subdivides into three different classifiers, each one associated with one procedure. This approach allows the specialization of each model in a particular procedure. On the other hand, the usage of only one classifier, which generalizes for the three procedures, could be a wrong choice due to the complexity of the problem. The usage of three classifiers fits well in this problem, because it's required a concrete answer to each procedure, with one classifier the solution mixes the three procedures description, which can cause some confusion

in the operator. Each classifier activates different class labels for the same input, due to the multiple steps to perform for some products. This classification problem fits in the multi-label classification, which is a generalization of the multi-class classification, that can assign various labels to the same instance.

The classification models tested for the procedure prediction were the Decision Tree (DT) and the K-Nearest Neighbors (KNN). Decision Trees (DT) are an intuitive and straightforward method, supported in both single-label and multi-label classification problems. The algorithm uses a tree structure where the internal nodes are conditions related to the features; the condition answer splits into branches, that could be other conditions or in case of end node the decision. Other of the algorithms that allow multi-label classification is the K-Nearest Neighbors (KNN), which groups the data in the more similar N clusters using the distance between instances. As model hyper-parameters, the DT classifier uses the default values of the *scikit-learn* framework; the KNN classifier uses 15 k-neighbors, which is a proximal value to the number of labels.

#### E. Description Reconstruction

The description reconstruction component is critical, considering the operator must understand the predicted procedure. The output of the classifier is an array of tuples, which usually it's hard to know by the operator. To the operator understand this information clearly, it's added an extra step that receives the classifier output and transforms it into a standard sentence comprehensible by the operator.

The easiest way to perform this task was re-using the description sentences that generate the class labels. So, during the process of label extraction, from the description of the procedure, it was associated with each label the most common sentence that generates it. So, the result is an association between a sentence (step description) and the relation tuple (classifier label), that maps the classifier output to a clean procedure step description. Even though the simplicity of this approach, it allows the description reconstruction, maintaining the generalizing capabilities of the label extractor.

### IV. EXPERIMENTS & RESULTS

After describing the architecture and mechanisms of the MSDS-OPP tool, it's time to perform a set of tests to validate those functionalities and compare them with the existing ones. Initially were evaluated the three procedures predictors, testing different classification models for each one. The metrics used to assess the different classification models are the recall, the precision, and the F1-score. Additionally, it was measured the area under the ROC curve (AUC), to confirm the previous metrics. The second study focus on an end-to-end analysis of the fire fighting procedures predictor. The process was analyzed, starting with the features and labels analysis, building a bag-of-words model, allowing the visualization of the total number activation in both features and labels. Regarding, the unbalanced data, in particular, the output class labels, Figure 4, the evaluation of the tool should follow some constraints, to

avoid corrupted results. Given that, the usage of more relevant performance metrics, such as the precision, recall, F1-score and AUC, would reflect the correct training of the model.

#### A. General Results

The overall results have the aim to evaluate each classifier's performance. For that, the considered performance metrics were the precision, the recall, the F1-score, and the area under the ROC curve (AUC). Due to the large dataset size, for the evaluation purposes, only 10% of them it's required, which corresponds to 23000 MSDS files. From the partial dataset, 70% was for training the algorithms, and the rest 30% for the test set. Figure 3 shows the obtained results for the three procedures predictors (fire, disposal, and storage), where each one was validated using two different classification models, the Decision Tree (DT) and the K-Nearest Neighbors (KNN). The graph on top shows the comparison between the F1-score and the area under the curve (AUC); the bottom graph compares the precision and recall.

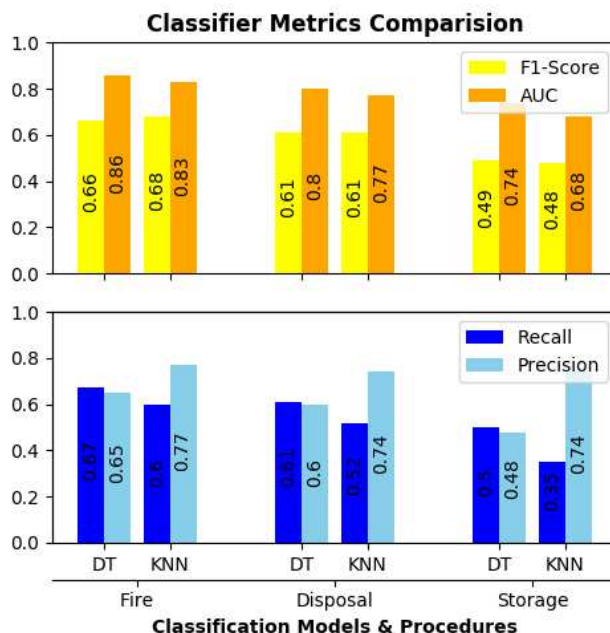


Fig. 3. Classifiers Comparison

The graph analysis shows that the procedure with more accurate predictions is the fire fighting predictor, and the worst is the storage predictor. This discrepancy in the results reflects the complexity associated with each procedure prediction. Due, the typical steps associated with the correct procedures, e.g., the phrase "dispose in accordance with the federal laws" is one of the most commons steps to perform according to the disposal procedures. Regarding the constraint, the overall results are suitable to the problem complexity, with an F1-score near to 0.65 and an AUC of 0.75. Even though these values are far from 1, they are adequate to the complexity of the class labels and the problem itself.

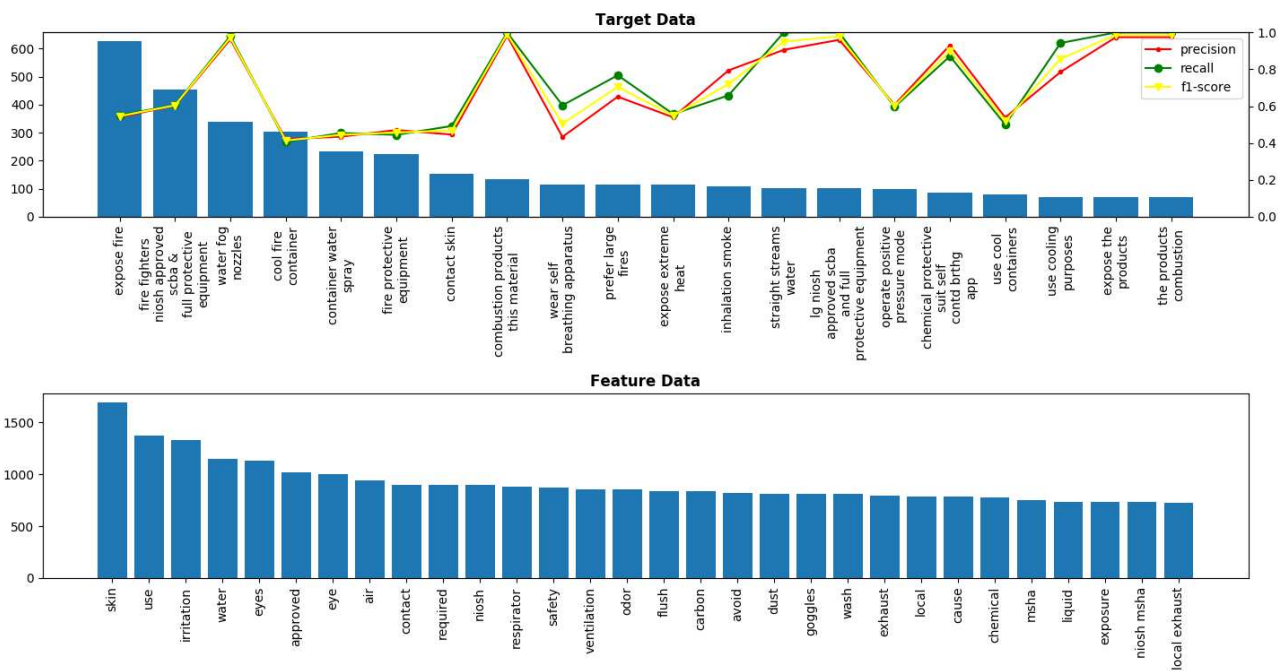


Fig. 4. Fire Fighting Procedures Classifier Features and Labels Bag-of-Words

Focusing in the difference between the classification models performance, the overall results are pretty similar regarding the F1-score and the AUC. In terms of precision-recall analysis, the button graph shows that the Decision Tree algorithm has an equalized value of precision and recall. However, the K-Nearest Neighbors algorithm has more discrepant values with high values of precision and lower values of recall. These results reflect that the KNN algorithm classifies only the high probability true positives instances, ignoring the less probable true positives, which results in a more precise algorithm. Regarding that outcome, the classification model to use may depend on the situations if the operator wants a more precise set of steps to perform uses the K-Nearest Neighbors, otherwise if he wants an initial guess or a recommendation for the actions of the procedures uses the Decision Tree algorithm.

### B. Fire Fighting Results

The overall classification results only validate the tool superficially. Based on that knowledge, a more detailed study of the whole process would help to understand their gaps and constraints. That analysis considers the fire fighting procedures classifier that uses a Decision Tree as a classification model.

The first analysis to make is checking the output from the features extraction pipeline and the label transformation pipeline. Both are present in Figure 4, where the graph on the top has the 20 classes used by the classifier, and in the button, there is a bag-of-words representation of the words with highest weight used as features by the classifier. Also, the target data graph shows the performance metrics (precision, recall, and F1-score) associated with each class.

The performance metrics indicate that some of the classes have results near to 1.0, while the others have results around 0.5 and 0.6. Additionally, some of the class labels denote more complex relations (e.g., "full protective equipment" or "inhalation smoke") than others (e.g., "contact skin"). The class distribution is unbalanced, having the first classes more instances, which isn't a constraint due to the stable values of precision and recall. The data used for the bag-of-words representation was from the training set, and the metrics provide from the test set.

The feature data has a more explicit form, most of the time represented by one or two words. Concerning that, the features are standard for the three predictors; some of the terms have more weight depending on the predicted procedure (e.g., the tokens "water" and "ventilation" are more related to the fire fighting procedures prediction).

### V. CONCLUSIONS & FUTURE WORK

Summarizing, the MSDS-OPP allows a quick response to the steps of the procedures to perform by the operator in certain situations. The tool permits the automatic acquisition of MSDS's from the web, and extraction of essential components from those files, making more straightforward and reliable the information retrieval. This way, the user, operator or not, have an additional tool that helps in their decisions and improves their quality of work. Regarding the results, the MSDS-OPP performs with satisfactory values of precision, recall, F1-score, and AUC, using both classification models (DT and KNN). The usage of two classification models allow to specify if



he wants a more precise algorithm (KNN), returning only the more probable steps to perform, or a more balanced one (DT).

Nevertheless, this tool has its constraints; one of them is the duplication of information between the features and the targets, generating some bias in the results. As a partial solution, the process of information extraction from the MSDS was improved, making the regular expressions more accurate, and the class labels more precise. Although, the multiprocessing implementation, the MSDS-OPP can accelerate its predictions, optimizing mainly the label transformation component.

As future work, one of the goals is to predict more types of operator procedures (e.g., spill procedures), making the MSDS-OPP more reliable and generic for a significant number of situations. Also, finding patterns that associate the MSDS structure/writing style with possible errors would help to rectify potential issues like the flash point underestimation. In terms of implementation, it's good to validate different approaches, testing different classification models in the MSDS-OPP. Still, in the case of the text processing, there are some alternatives to be explored, like word embedding. Additionally, the procedures reconstruction uses a fundamental approach to perform this task, the utilization of other options, like autoencoders, could perform better.

## REFERENCES

- [1] J. Lee, B. Bagheri, and H.-A. Kao, "A cyber-physical systems architecture for industry 4.0-based manufacturing systems," *Manufacturing Letters*, vol. 3, pp. 18 – 23, 2015. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S221384631400025X>
- [2] A. Nicol, A. C. Hurrell, D. Wahyuni, W. McDowall, and W. Chu, "Accuracy, comprehensibility, and use of material safety data sheets: A review," *American Journal of Industrial Medicine*, vol. 51, no. 11, pp. 861–876, 11 2008. [Online]. Available: <https://doi.org/10.1002/ajim.20613>
- [3] J. A. Bernstein, "Material safety data sheets: Are they reliable in identifying human hazards?" *Journal of Allergy and Clinical Immunology*, vol. 110, no. 1, pp. 35 – 38, 2002. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0091674902000374>
- [4] M. I. Greenberg, D. C. Cone, and J. R. Roberts, "Material safety data sheet: A useful resource for the emergency physician," *Annals of Emergency Medicine*, vol. 27, no. 3, pp. 347 – 352, 1996. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S019606449670272X>
- [5] D. Radnoff, "Accuracy of reported flash point values on material safety data sheets and the impact on product classification," *Journal of Occupational and Environmental Hygiene*, vol. 10, no. 10, pp. 540–546, 2013, pMID: 24011179. [Online]. Available: <https://doi.org/10.1080/15459624.2013.818233>
- [6] C. C. Phillips, B. C. Wallace, C. B. Hamilton, R. T. Pursley, G. C. Petty, and C. K. Bayne, "The efficacy of material safety data sheets and worker acceptability," *Journal of Safety Research*, vol. 30, no. 2, pp. 113 – 122, 1999. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0022437599000055>
- [7] G.-H. Cornelia, "Searching for hazardous substances on the internet," vol. 25, no. 4, pp. 257–266, Jan 2001. [Online]. Available: <https://doi.org/10.1108/EUM0000000005743>
- [8] M. Pauwels and V. Rogiers, "Database search for safety information on cosmetic ingredients," *Regulatory Toxicology and Pharmacology*, vol. 49, no. 3, pp. 208 – 216, 2007. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0273230007001146>
- [9] G. Jang, T. Lee, S. Hwang, C. Park, J. Ahn, S. Seo, Y. Hwang, and Y. Yoon, "Piston: Predicting drug indications and side effects using topic modeling and natural language processing," *Journal of Biomedical Informatics*, vol. 87, pp. 96 – 107, 2018. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1532046418301898>
- [10] T. Ly, C. Pamer, O. Dang, S. Brajovic, S. Haider, T. Botsis, D. Milward, A. Winter, S. Lu, and R. Ball, "Evaluation of natural language processing (nlp) systems to annotate drug product labeling with meddra terminology," *Journal of Biomedical Informatics*, vol. 83, pp. 73 – 86, 2018. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1532046418301060>
- [11] J. A. Reyes-Ortiz, B. A. González-Beltrán, and L. Gallardo-López, "Clinical decision support systems: A survey of nlp-based approaches from unstructured data," in *2015 26th International Workshop on Database and Expert Systems Applications (DEXA)*, Sep. 2015, pp. 163–167.
- [12] N. Shanavas, H. Wang, Z. Lin, and G. Hawe, "Structure-based supervised term weighting and regularization for text classification," in *Natural Language Processing and Information Systems*, E. Métais, F. Meziane, S. Vadera, V. Sugumaran, and M. Sarace, Eds. Cham: Springer International Publishing, 2019, pp. 105–117.
- [13] Z. Zhen and J. Zhang, "Text classification based on latent semantic indexing and graph embedding," in *2018 14th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, July 2018, pp. 1001–1004.
- [14] N.-W. Chi, K.-Y. Lin, and S.-H. Hsieh, "Using ontology-based text classification to assist job hazard analysis," *Advanced Engineering Informatics*, vol. 28, no. 4, pp. 381 – 394, 2014. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1474034614000354>
- [15] A. E. . P. Corp and I. Tracor Jitco, *Registry of toxic effects of chemical substances*. US Department of Health and Human Services, Public Health Service, 1980.
- [16] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer Networks*, vol. 30, pp. 107–117, 1998. [Online]. Available: <http://www-db.stanford.edu/backrub/google.html>
- [17] R. Zhao, G. Li, J. Liu, and X. Wang, "Clinical multi-label free text classification by exploiting disease label relation," in *2013 IEEE International Conference on Bioinformatics and Biomedicine*, Dec 2013, pp. 311–315.
- [18] B. Parlak and A. K. Uysal, "Classification of medical documents according to diseases," in *2015 23rd Signal Processing and Communications Applications Conference (SIU)*. IEEE, may 2015.
- [19] J. Nam, J. Kim, E. Loza Mencía, I. Gurevych, and J. Fürnkranz, "Large-scale multi-label text classification — revisiting neural networks," in *Machine Learning and Knowledge Discovery in Databases*, T. Calders, F. Esposito, E. Hüllermeier, and R. Meo, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 437–452.
- [20] 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 E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [21] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc., 2009.
- [22] L. Richardson, "Beautiful soup documentation," 2007.

# Offensive assessment in social networks

Carlos Guilherme Fontes Tomás

ProDEI

Faculty of Engineering, University of Porto

Rua Dr. Roberto Frias, 4200-465 Porto, Portugal

up201902801@fe.up.pt

**Abstract**—In this paper, several architectures are trained to obtain the best possible results in the task of classifying a tweet as offensive or non-offensive. The architectures used are based on Random Forest, Support Vector Classification, Naive Bayes Classifier, Voting, Linear Regression, K-nearest Neighbors, Gradient Boosting, decision tree, and a more advanced one Bert. The deep learning Bert model outperformed the other models, achieving accuracies of 85.5% and f1 of 82.2%. Sentiment analysis was made, when added to the dataset, allowed to obtain better classification results.

**Index Terms**—Text mining, Bert, Offensiveness

## I. INTRODUCTION

A social network is a website or application which enables users to socialize with each other by posting information, comments, messages, images, and others. The benefits are many, a family in different corners of the world can get together in seconds. It may be possible to find friends and get in touch; it is possible to create events, grow a business, one can occupy free time with funny videos and news, it is possible to share opinion and art, it is possible to make new friends (fight loneliness) and meet people from common interests. In recent years with the proliferation of social networks, conditions have emerged that allow users often protected by anonymity to say what they want without filters, sometimes becoming offensive. Problems such as hate speech and cyberbullying spread on the internet, causing hate to grow, often spreading to the physical world, bringing severe consequences. Then comes the task of removing offensive or inappropriate content, manually filtering responsibility of such content becomes time-consuming and may cause post-traumatic stress symptoms to reviewers and the amount of new content each day is enormous, there have been studies to try to automate this task. The task is usually defined as a supervised classification problem, where systems already trained, searching for the presence of some form of abuse or offensive content. The task, OffensEval19 A, uses the Offensive Language Identification Dataset (OLID), which is annotated following a hierarchical three-level annotation schema that takes both the target and the type of offensive content into account.

In Section II will be presented the data and how the data is prepared to be used for each algorithm. In Section III will be presented all the data handling and processing. In Section IV will be presented the learning models used in this work and metrics used for evaluation. In section V will be presented with the results obtained. In section VI will be presented

the conclusions. Finally Section VII the future work will be presented.

## II. DATA

Training data for the competition [1] was given in a plain text format of the table separated values consisting of tweet ID and tweet content, with three labels values for task A = OF (offensive), NOT(not offensive) the objective is a classification task where it is necessary to identify whether the tweet is offensive or not, task B = TIN(Targeted Insult and Threats), UNT(Untargeted) automatic categorization of offense types, and task C = IND(Individual), GRP(Group), OTH(Other). The values for the values for tasks B and C offense target identification were discarded. Before training the models, the data was preprocessed extensively. The primary aim of this is to ensure that models were trained on the most normalized representation of tweets possible, allowing them to take full advantage of the data, with the overall goal of increasing model performance as much as possible. In Fig. 1 a small example of the dataset is presented

id	tweet	subtask a	subtask b	subtask c
86426	@USER She should ask a few native Americans wh...	OFF	UNT	NaN
98194	@USER @USER Go home you're drunk!!! @USER #M...	OFF	TIN	IND
16820	Amazon is investigating Chinese employees who ...	NOT	NaN	NaN
62688	@USER Someone should've taken this piece of sh...	OFF	UNT	NaN
43605	@USER @USER Obama wanted liberals & illeg...	NOT	NaN	NaN

Fig. 1. HEAD dataset

As shown in Fig. 2, the data is unbalanced. There are more than twice as many not offensive tweets (8840 vs. offensive tweets 4400).

## III. DATA PREPARATION

This section outlines all the data handling and processing techniques carried out.

### A. Hashtag

Hashtags can contain relevant information, it is vital to extract the most appropriate words, for this is used the segmentation library python **wordsegment** [2].

### B. Emoji

Emoji can contain relevant information like how the person is feeling and what the person thinks about the situation, so it is essential to extract the emojis. For extract, emojis is used library **Emoji for Python** [3].

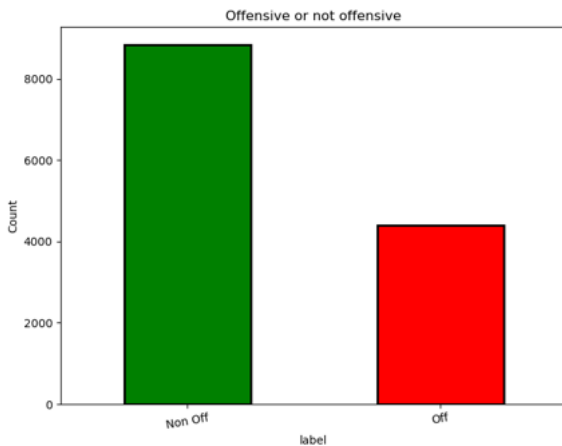


Fig. 2. HEAD dataset (Offensive and not offensive) label distribution

### C. Text to Vector

Since the primary source of information of tweet post contains only text and it is hard to use the text itself to categorize, the text is transformed in feature vectors. Feature vectors can be created using several methods like Bag of Words (**BoW**), Term Frequency Inverse Document Frequency (**TF-IDF**), and others. The **TF-IDF** measure allows us to evaluate the importance of a word(term) to a twitter post. The importance is proportional to the number of times a word appears in the post but is offset by the number of posts in the dataset that contain the word, which helps to adjust for the fact that some words appear more frequently in general. To obtaining better results with **TF-IDF**, **GridSearchCV** library from sklearn is used with multiple parameters to get the best combination for classification. The **TF-IDF** gives better results than bow maybe because the tweets are not well written or grammatically correct.

### D. Synonyms

Synonyms are words that have the same meaning. Then a synonym list was created if the word exists in the list, then nothing is done if the word does not exist, then looks for its synonyms if any of it exists in the list then it is replaced with its synonym in the tweet post if not found the word is added to the list. With this process, it was intended to ensure that the meaning of the words is maintained independently of the exact form displayed in the sentence, allowing to increase the performance of the model. To get the synonyms library, **wordnet** from **nltk** [4] is used.

### E. Text edits

Stop words and punctuation were removed, all text was converted to lowercase, lemmatization and porter stemmer was applied, text smaller than one character was removed, TF-IDF was applied so the sentence can be converted to feature vectors that can be used as input to estimators.

### F. Violin plots

For columns added by sentiments, bad words and pos tagging, visualizations were generated using violin plots

Violin plots have many summary statistics. The white dot represents the median. The thick gray bar in the center represents the interquartile range, and the thin gray line represents the rest of the distribution, except for points that are determined to be “outliers.” On each side of the gray line is a kernel density estimation to show the distribution shape of the data. More extensive sections of the violin plot represent a higher probability that members of the population will take on the given value; the skinnier parts represent a lower probability.

### G. Sentiment

Sentiment analysis was performed to understand what kind of positive, negative or neutral feelings are present in the tweets, as greater negativity can mean a more offensive tweet. To obtain these values are used the **vaderSentiment** library [5] with the input of post tweets. Four columns are added to the dataset (which had already been processed by **TF-IDF**) the counts **vader\_neg** (negativity), **vader\_pos** (positivity), **vader\_neu** (neutrality) and **vader\_compound** (a general value ranging from -1 extreme negativity to 1 extreme positivity). When analyzing the graphs in figures 3 to 6, there seems to be a relationship between the text’s negativity/positivity and offensiveness. Verifying that the mean of non-offense is close to the center of the graph close to neutrality and the offense finds its average center close to -0.4 already in the negativity.

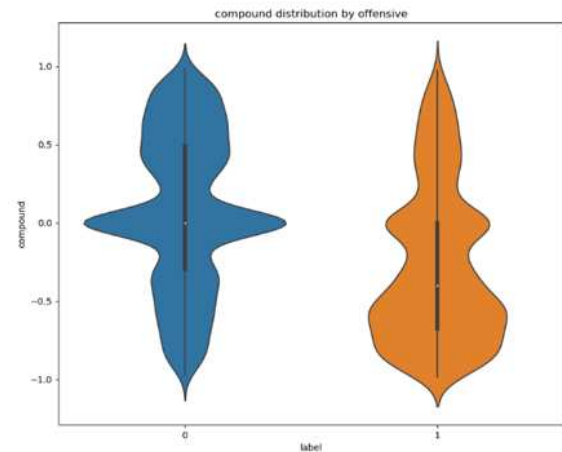


Fig. 3. Compound Distribution

### H. Bad Words

The **offensive wordlist for YouTube** [6] is used to create a new column in the data set named badWords. There are words that, by default, will make a post more offensive often. These words are disguised with some special characters and

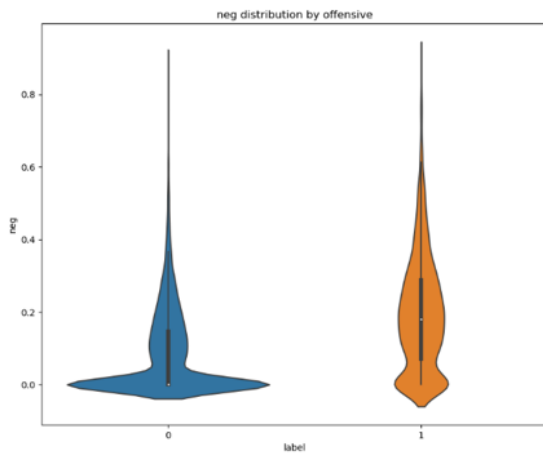


Fig. 4. Neu Distribution

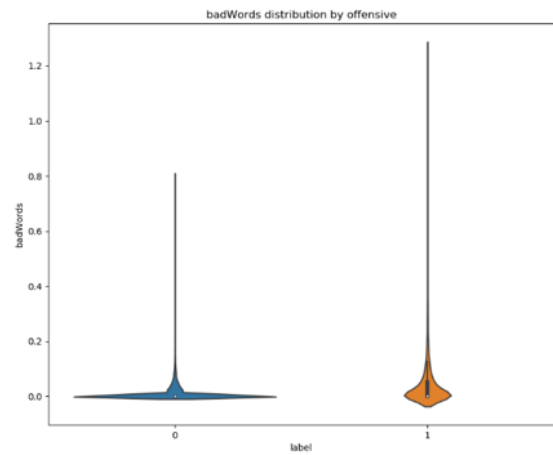


Fig. 6. Bad Words Distribution

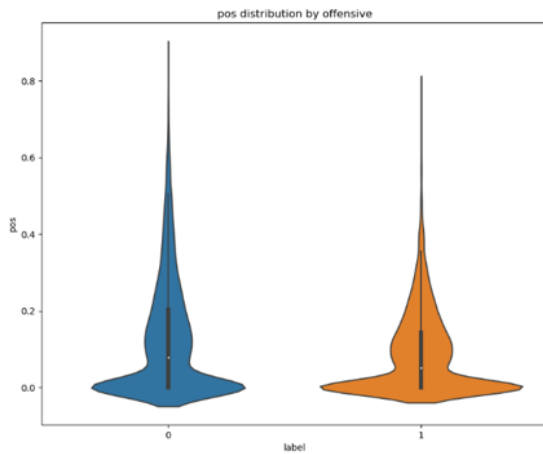


Fig. 5. Pos Distribution

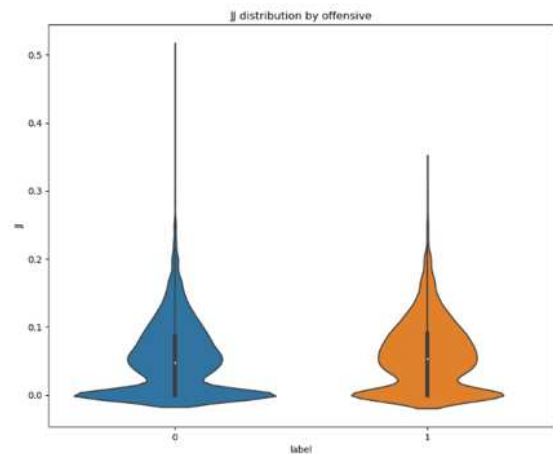


Fig. 7. JJ Distribution

spelled ambiguously or differently so that filters will not detect them. With this technique, it is intended to emphasize these words for the model, so that they do not pass unnoticed. When analyzing the graph in Fig. 6, it seems that the existence of bad words in the posts has some relation with the text being more offensive.

### I. Part-of-speech tagging (POS tagging)

The `nlTK` library was used to detect the number of adjectives, names, and verbs in each tweet as they may have some correlation with whether a sentence becomes offensive or not. An offensive phrase is expected to have many adjectives. When analyzing the graphs of Figures 7 to 9, it is not clear that the lexical classification of words has any relation to the offensiveness of the text.

### J. Pipeline

The data is imported into a data frame using the `pandas` library [7], then lines with errors or with empty values are removed, columns belonging to tasks b and c are eliminated, the label from task A is mapped to ('NOT' (no offensive) = 0, 'OFF' (offensive) = 1) and emojis and hashtags are extracted. The previous columns will be augmented by four more columns from the sentiment analysis. After the tweet post is **POS tagged** and the number of adjectives, verbs and names counted, three new columns added each is representing its word class. Then the text column is filtered by the YouTube bad word list, and a new column with this value added to the data frame. The next step taken is to use the text column to get vectors. First, the text is tokenized, single characters, punctuation, and stop words removed. Then the words " 't,

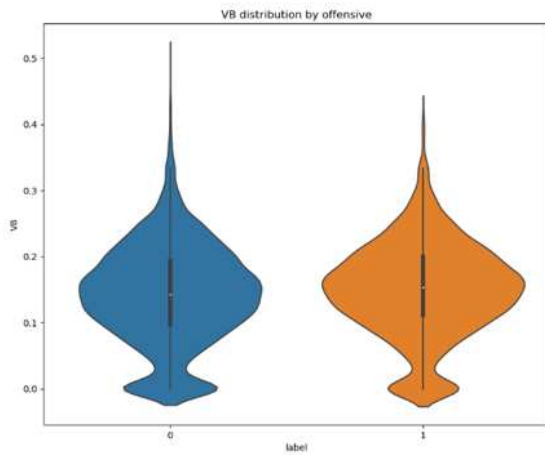


Fig. 8. VB Distribution

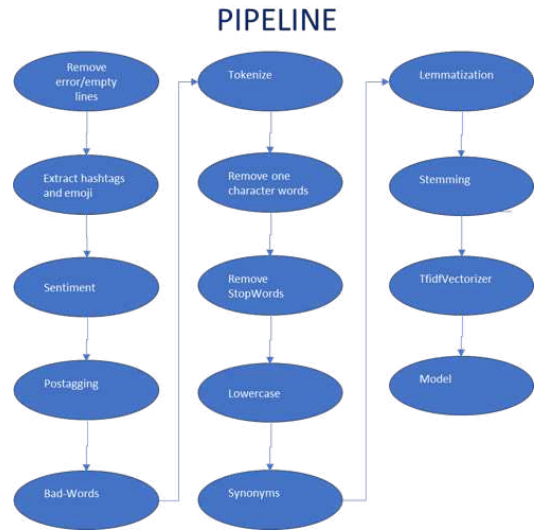


Fig. 10. General Pipeline

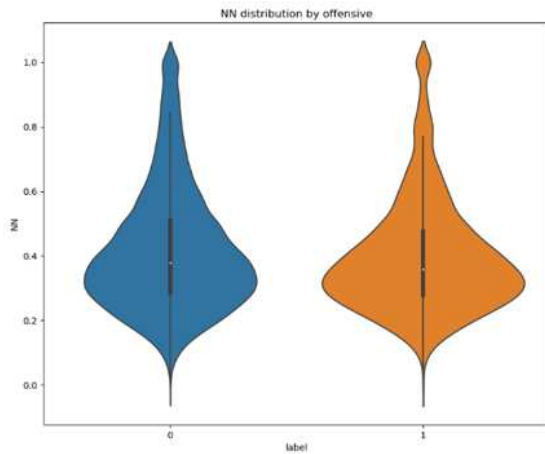


Fig. 9. NN Distribution

## PIPELINE BERT

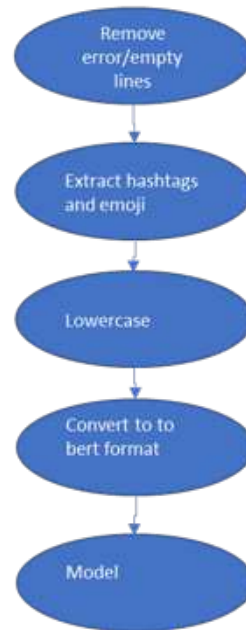


Fig. 11. BERT Pipeline

not, nor and no” are extracted and will be kept as they have a negation/negativity effect is essential to keep this in the data frame. This process followed by lemmatization and stemming. The result of the previous operation is given to the **TF-IDF** algorithm that will prune accents and all lowercase text and will return an array of vectors concatenated with the columns added to the data set to be processed by the models.

To decide which are the best input parameters in **TF-IDF**, a **GridSearchCV** used in a pipeline with **LinearSVC**.

The data processed for the Bert algorithm (see next Section) consisted only of extracting emojis and hashtags and setting the data processing mode to "Bert." No words removed. No lemmatization and stemming were applied as these operations returned worse results in the case of this model.

## IV. METHODS AND EVALUATION

The following algorithms, Random Forest, Support Vector Classification, Naive Bayes Classifier, Logistic Regression, and k-nearest neighbors created using a scikit-learn library [8]. Bert (Bidirectional Encoder Representations Transformers) used from the library ktrain [9]. The final evaluation made using a different data set that is processed like the original dataset(used for training) to get the same properties, and this

TABLE I  
FINAL RESULTS FOR ACCURACY, F1 MACRO AVG, PD, PF, PRECISION,  
KFOLD ACCURACY AND KFOLD F1

[HTML]COCOCO	Acc	F1	PD	PF	Prec	Kf Acc	Kf F1
[HTML]COCOCORandom Forest	0.816	0.729	0.811	0.183	0.446	0.758	0.684
[HTML]COCOCOSVC	0.812	0.753	0.693	0.152	0.583	0.755	0.709
[HTML]COCOCONaive Bayes Classifier	0.791	0.649	0.905	0.220	0.279	0.733	0.612
[HTML]COCOCOLogistic Regression	0.827	0.753	0.805	0.169	0.5	0.769	0.707
[HTML]COCOCOK-nearest neighbors	0.724	0.528	0.523	0.259	0.142	NV	NV
[HTML]COCOCOBert	0.855	0.822	0.729	0.094	0.762	NV	NV

dataset has ‘not offensive’ tweets 620 vs. ‘offensive’ tweets 240.

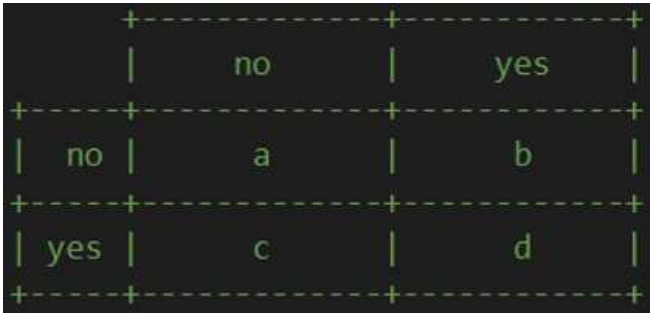


Fig. 12. Confusion Matrix

The metrics for the evaluation of the model, are presented in Fig. 13.

Accuracy	= Acc	= $(a+d)/(a+b+c+d)$
F1	= F1	= $2 * (1/((1/Precision)+(1/recall)))$
Probability of detection	= pd = recall	= $d/(b+d)$
Probability of false alarm	= pf	= $c/(a+c)$
Precision	= prec	= $d/(c+d)$

Fig. 13. Metrics for evaluation from the confusion matrix in Fig. 12

## V. RESULTS

In Table I is possible to observe that Bert provides the best results in the test dataset, outperforming the scikit-learn algorithms by a decent margin. The Logistic Regression is the second-best model and outperforms other scikit-learn algorithms. The models do not obtain better results on the test set, due to the lack of data and unbalanced data in the training set, which does not allow them to learn to distinguish between classes at a high level of confidence. To minimize the effects of this situation, a data augmentation technique that consists of adding new data lines where the text is changed by replacing words by its synonyms were used but did not get good results.

The use of synonyms appears to enhance the model’s result because by limiting the text to synonyms, it is more likely that an unknown word will change to a word that exists in the training dataset, causing the test dataset to have a greater relationship with it. The reason that BERT has obtained results that are comfortably superior to the other models seems to be due to its pre-training, which allows a transfer of knowledge and also to the fact of its bidirectionality.

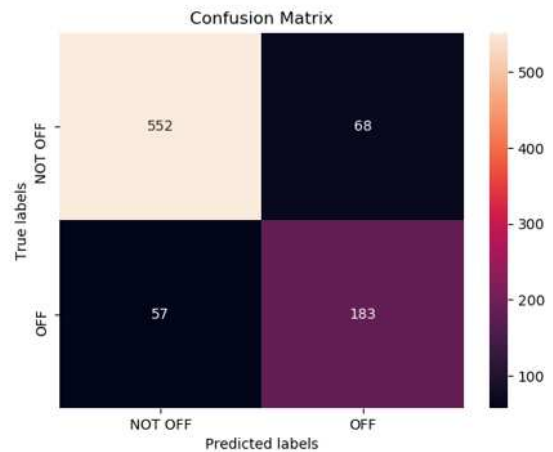


Fig. 14. Confusion matrix for BERT

## VI. CONCLUSIONS

Classify/extract text information is a complex task when dealing with social networking text that is wrong writing and often has hidden meanings in hashtags and emojis that can transform the entire sense of a sentence/post, so it is essential to extract some meaning of these elements to get better results in text extraction. In this particular task, we also find better results by adding the sentiment analysis columns. It appears that greater negativity also represents greater offensiveness. The Bert algorithm was able to achieve better results than the other algorithms, although no sentiment analysis used in this case.

## VII. FUTURE RESEARCH

In future works, it would be interesting to use the Bert model result as a column of the data frame, in combination with columns obtained in Subsections III-G and III-H, and then use this with the Logistic Regression algorithm and verify the result. This tool can also be used in the context of video and image to process the given speech after transformed into text.

## REFERENCES

- [1] M. Zampieri, S. Malmasi, P. Nakov, S. Rosenthal, N. Farra, and R. Kumar, “Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval),” *arXiv preprint arXiv:1903.08983*, 2019.
- [2] “wordsegment.” [Online]. Available: <https://pypi.org/project/wordsegment/>
- [3] T. Kim and K. Wurster, “emoji.” [Online]. Available: <https://pypi.org/project/emoji/>.
- [4] S. Bird, E. Loper, and E. Klein, “Natural language processing with python o’reilly media inc,” 2009.
- [5] C. J. Hutto and E. Gilbert, “Vader: A parsimonious rule-based model for sentiment analysis of social media text,” in *Eighth international AAAI conference on weblogs and social media*, 2014.
- [6] J. Parker and S. Zen, “Youtube blacklist words list - youtube comment blacklist,” Apr 2019. [Online]. Available: [https://www.freewebheaders.com/youtube-blacklist-words-list\\_youtube-comment-moderation/](https://www.freewebheaders.com/youtube-blacklist-words-list_youtube-comment-moderation/)

- [7] W. McKinney, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*, S. van der Walt and J. Millman, Eds., 2010, pp. 51 – 56.
- [8] F. Pedregosa and G. a. a. Varoquaux, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.
- [9] S. A. Amaiya, "amaiya/ktrain," Feb 2020. [Online]. Available: <https://github.com/amaiya/ktrain>

# Finding Emotions in Screenplays for Genre Classification

Inês Nunes Teixeira

Faculty of Engineering of the University of Porto FEUP

INESC TEC

Porto, Portugal

up201104124@fe.up.pt

**Abstract**—Last century’s history can be exposed by several means, including cinema. Millions of films, spread in the world, provide relevant content for artistic analysis contributing to understanding the evolution of society. However, most of them are stored in archives, undocumented. Big data manual annotation, which would enable retrieving and semantically relating contents, is a costly and time-consuming task.

This paper proposes a first analysis of this problem by providing an approach for film genre detection using script emotion analysis. Firstly, several lists of dictionaries were created for identification of specific types of emotions. Secondly, information extraction techniques were implied for both automatic annotation purposes and artistic cinematographic analysis of movie scripts. This serves as a contribution in the automatic annotation of content and finding relevant artistic characteristics in films. Lastly, and mainly, this approach aims to prove the existence of a pattern in the use of emotion-related vocabulary in movie scripts and the genre classification of the film.

The achieved results were very promising since it was shown that there are considerable similarities between emotion distribution in films from the same genres.

**Index Terms**—information extraction, exploratory data analysis, emotion detection, genre classification, data visualization

## I. INTRODUCTION

There is much research in the film analysis field nowadays, developed in both the arts and the social sciences communities and, in the last few years, different approaches have been developed, inspired by film literature. This raised the awareness of the research potential created to the multimedia technologies field to apply innovative solutions to an important and popular area of entertainment and expression: the cinema.

The main purpose of this paper represents a first step into automatic annotation of cinematographic contents, as both a data annotation contribution and the production of useful data in the artistic analysis of cinematic work. Analysing the distribution of emotions in screenplays is a low-level analysis of content, that can be used for high-level film details inference, like automatic classification of film genre. This can be a significant contribution in cinema research, as the analysis of big amounts of data enables pattern recognition for finding relationships between contents.

Although this is an extensive field, this study focuses specifically in analysing screenplays for information extraction, to understand if a film from a specific genre has a different set of vocabulary, describing a specific set of emotions. For

this study, a dataset with a thousand screenplays extracted from The Internet Movie Script Database (IMSDb) [16] was used. The scripts contain the film title and classified genres (according to IMSDb classification), and all the dialogues and scene descriptions, so these contents are naturally more detailed on settings than dialogue transcripts. In Figure 1, the dataset distribution is presented for genre classification.

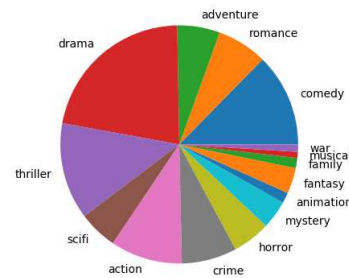


Fig. 1. Pie chart for genre distribution of used dataset.

Using this data, a pre-processing module is required for token extraction and removal of stop words and punctuation. Afterwards, the detection of emotions is a complex task, so several approaches have to be tested to analyse patterns in vocabulary distribution. Here, data visualisation techniques provide a more straightforward analysis means. Lastly, using the emotion vocabulary distribution, it should be possible to use a supervised intelligent system for genre classification, using the pre-annotated genres by IMSDb [16] as ground truth.

As such, this document first presents a study on genre classification fields and emotion extraction. Also, some analysis of the data visualisation field is presented. The related work overview is presented in Section II. For defining the approach, some context on basic emotion classification and visualisation is discussed in Section III. Section IV presents the description of the approach for emotion extraction, Section VI explores data visualization, whereas the genre classifier is summarily described in V. Lastly, the final results are discussed in VII, and some conclusions of this approach are presented in VIII.

## II. RELATED WORK

In order to analyse the state-of-the-art for similar approaches presented recently in the scientific community, this section



presents an overview in two fields: related work in automatic genre classification and emotion analysis in text, as text processing and information extraction approaches. Also, as a means to visualise the results, a brief overview of data visualisation is presented.

#### A. Automatic Genre Classification

Recently, several film genre classification approaches have been proposed. Most methods are based on visual features analysis, predicting the film genres by low and high-level features.

Genre detection with low-level features is presented in [1], proving that there is a relationship between visual information (without using audio and text cues) and genre classification. Four low-level features are exploited: shot length, shot motion content, lighting key and colour variance.

A scene categorization method is suggested in [2] applied to movie trailers. It represents trailers with a bag-of-visual-words and uses classes of vocabulary for each considered genre: action, comedy, drama and horror. This method does not consider dynamic components of scenes, i.e. the action and movement between shots, and has a limitation in approaching black and white films. However, it improved the classification performance over the use of low-level visual features alone.

More recently, [3] proposes using Artificial Intelligence for genre classification. It uses well-known image descriptors to compute high-level features for each keyframe. A CNN trained with both visual and audio features is used. Variation of information with time is, however, not considered. Even so, it achieves a 70% accuracy, claiming to outperform other state-of-the-art trailer classifiers: Gist [4] - developed to detect high-level characteristics of a scene; Centrist [5] - visual description for recognising natural scenes and indoor environments; w-Centrist [2] - based on Centrist but considering colour information; low-level feature analysis presented in [1].

Smart Trailer [13] proposes using the film's script both for genre characterization, as well as for scene selection and trailer production. Although focusing on video summarization, the approach presents good results on genre classification.

#### B. Emotion Detection

Besides genre classification, [1] provides an extensive analysis of films as a way of expression and highlights evidence of a set of rules used in film making able to transmit emotions and perceptions as described in the literature [6]. [7] also discusses this idea in depth.

Film psychology is another concern on video analysis, i.e. the emotion that a film provokes on its audience. In both mainstream and art-house cinema, a film provokes emotions on its viewers, and there are several studies on how a filmmaker can translate desired reactions and feelings on film metrics [20].

One of the most used strategies is in using close-ups of characters to emphasise emotion, which is related to facial expression recognition (FER) [21] [22] and natural datasets creation [23] [24]. Namely [22] approaches this with a new

CNN for detecting seven facial expressions and has very promising results.

Although cinema lives in an audiovisual world, as stated by Michael Schilf, "Film is Visual: Show, Don't Tell!" [8], there are several notes for emotions that can be observed in film screenplay. These notes are usually meant to give an idea of the emotion intend on each scene. Nevertheless, it was not found research work in the analysis of these emotional notes in screenplays in the present state-of-the-art.

Emotion extraction is a field of information extraction and can be viewed as a complex evolution of sentiment analysis. As a natural language processing task, sentiment analysis, or opinion mining, is an extensively studied field, showing important and useful results in marketing and advertising [8] [9], in recommendation systems, earlier in [10] and more recently in tourism reviews [11], while being still an evolving and exploring field [12].

Sentiment analysis is usually referred to as distinguishing positive from negative contents. To analyse in more detail what are the emotions associated with each content, i.e. if the content is negative because it provokes anger or sadness, the approaches are commonly categorised in the emotion extraction field. Although not so thoroughly explored, most of the state-of-the-art emotion extraction from text approaches are applied in the social networks field, using post contents such as text, hashtags and emoticons [14]. Unfortunately, there is still much lack of research in other applications, because of the complexity of the task, associated with three main difficulties: the complex nature of expressing emotion in text, translating in intensity, irony and variety in vocabulary depending on the context; the shortage of quality data, required to improve machine learning implementations; the inefficiency of the current models, as the commonly used multiclass emotion classifiers are either designed as one classifier for each emotion or one classifier for a pair of opposite emotions [15].

#### C. Simple approaches on Data Visualization

Technological advances created the opportunity for making available tremendous amounts of audiovisual content, otherwise hidden to small communities of privileged users. This brings excellent opportunities for entertainment, cultural enrichment, training, but also raises important challenges to enable making these resources effectively available.

Within this scenario, data visualization is one of the emerging subjects in state-of-the-art developments [25] [26] [27] [28], discussed to enable big data search, visualization and rendering, usually with recommendation systems incorporated. Research lines focus on identifying new paradigms for presenting the results retrieved from large assets, using more appealing and efficient concepts, rather than long lists of outputs. Achieving this aim requires being able to filter content, defining metrics to identify degrees of similarity, and how close to each other the individual elements are.

Therefore, in the specific task of emotion distribution analysis in screenplays, it is of utmost importance to explore visualisation techniques for better understanding patterns for

genre classification. In tasks of exploratory data analysis, visualisation tools can play an important role for this reason [15].

Specifically in audiovisual data representation, several recent work has presented interesting creative results [29], such as data visualization in connected networks [30] [31], social networks distributions [32], character interaction [33] and visual storylines [34] in films, and colour distribution [35]. One of the most referenced approaches in state-of-the-art is the Cinematics circles [36], where a film is represented in a ring-shaped chart, with colour distribution, scene cut and motion represented.

### III. PSYCHOLOGY OF EMOTION

To fully understand the problem of emotion extraction, it is a prerequisite to present the general idea of emotion theories in psychology. In fact, there is an enormous amount of different emotions, and analysing all of them would probably not be an achievable task for the time being. As such, there is the need of defining what are the main emotions that could be analysed in the setting of this approach. Although there is no universally accepted model of emotions, there are two main models used widely in both psychology and information extraction.

The Colour Wheel [17] presents eight primary emotions: Joy, Trust, Fear, Surprise, Sadness, Disgust, Anger and Anticipation. On the other hand, Ekman's study [18] revealed that there are seven basic international emotions: Joy, Anger, Fear, Sadness, Disgust, Contempt and Surprise. Both models are widely accepted and have been used in emotion detection literature [15].

There is also some research work present in the state-of-the-art on finding a relationship between colours and emotions

As it can be observed in Figure 2 the Colour Wheel [17] has not only a series of vocabulary associated with each basic emotion represented but also some mapping between emotions and colours. This has also been a topic studied in the field of psychology, and despite not existing a fixed rule for representing emotions in colours commonly agreed by all authors, it is mostly accepted the association between sadness and the colour blue [19].

### IV. EMOTION DISTRIBUTION ANALYSIS

As explained in previous sections, the implementation of this approach follows a simple workflow, demonstrated in Figure 3.

The first analysis of data consists of reading the screenplay files and extracting tokens, by removing blank spacing, punctuation and transforming capital letters into lowercase letters, and using the Natural Language Toolkit in Python for tokenization. Although word variations should be discarded for, for instance, ignoring singular-plural differences, for the purpose of this task, stemming techniques could not be used because of the prefix and suffix influence on meaning. I.e., if the words unhappy and happy were to appear as tokens, if using stemming techniques, they would become the same

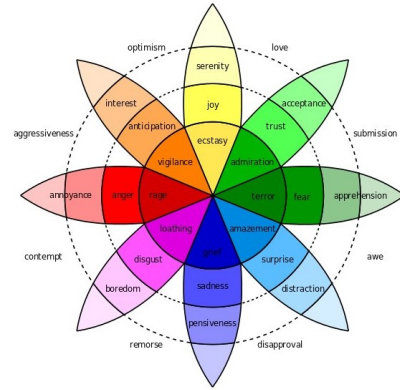


Fig. 2. Plutchik's Wheel of emotions [17]

token, as the “un” prefix would be ignored. In emotion-related tasks, this is not advised as the purpose is to understand if it would be a token related to joy, or on the contrary, related to sadness. Thus, by the end of this first model, a list of tokens per script was obtained and is used as input for the next two models of the workflow. These will be described in detail in the next subsections.



Fig. 3. System workflow representation.

#### A. Basic bag-of-words

In many approaches presented in II, the major difficulty is to decide how to define what words are related to each emotion. Using a dictionary of words could solve this problem, but each word is very dependent on the context. In other words, a word in a dictionary may be related to the emotion of joy in a particular context, but may also be used as irony in other circumstances. On the other hand, defining a list of words for each emotion can also depend on the goal of the approach, i.e., if using vocabulary for translating real-life feelings [37] [38], these might not apply to the cinematographic description of a scene. Ideally, for the purpose of analysing emotions in screenplays, a specific dictionary would be used, but no such data can be found in the literature.

As such, the first analysis of the data was based on a naive approach, by analysing the distribution of the words presented in Ekman's study [18]: Joy, Anger, Fear, Sadness, Disgust, Contempt and Surprise. The obtained results are represented in Figure 4, as a simple illustration where only the analysis of four genres is shown.

Although it is noticeable some different distributions in the vocabulary of each genre, the disparities are not evident.

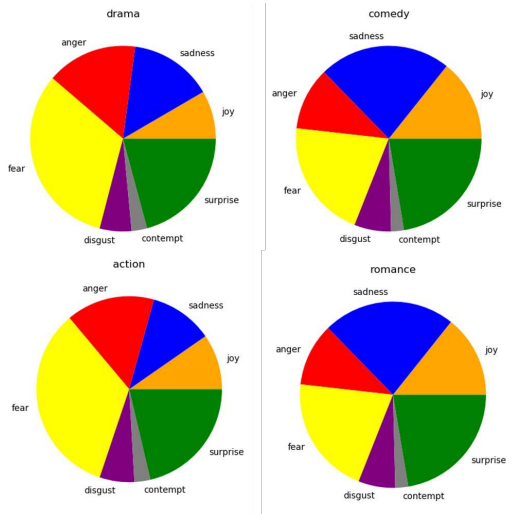


Fig. 4. Simple illustration of basic emotion vocabulary distribution in four types of films: Drama, Comedy, Action and Romance

### B. Synonyms bag-of-words

Instead of using only one word per emotion, to improve the previous results, using a dictionary of words was considered. Firstly, the words for each dictionary followed the vocabulary suggested in [38], where, for instance, joy can also be described by words like happy, optimistic, powerful, accepted and proud.

Unfortunately, this approach revealed results that were difficult to interpret because the emotion of Joy gained massive importance in every analysed genre, as shown in Figure 5.

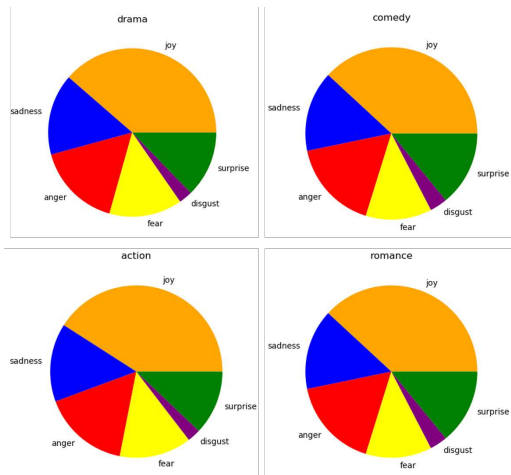


Fig. 5. Simple illustration of emotion vocabulary [38] distribution in four types of films: Drama, Comedy, Action and Romance

This fact can be due to the type of word considered as indicators for Joy, like “open” and “important”. These vocabularies can describe happiness as a feeling, i.e. if someone says they are feeling important. In other contexts, it could be a word independent of any of the analysed basic emotions.

This analysis is mostly evident in Figure 6, where the genre Thriller would not be expected to have such amount of words related to the emotion Joy.

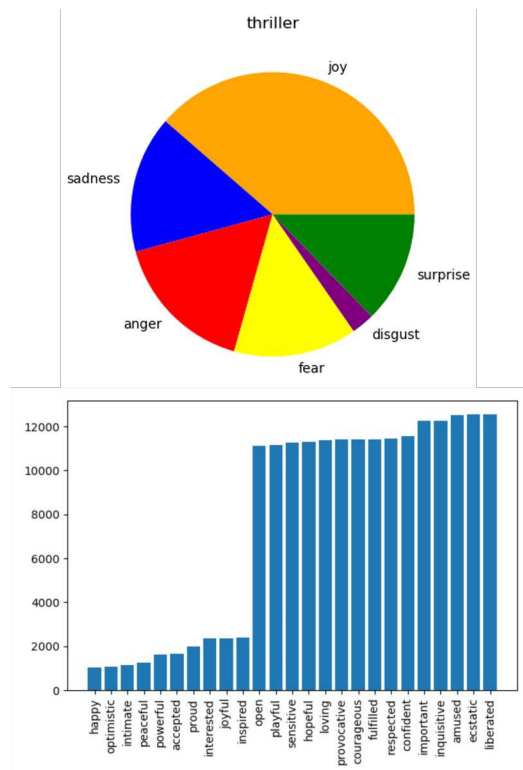


Fig. 6. On top: Illustration of emotion vocabulary [38] distribution in Thriller screenplays. On bottom: histogram of vocabulary distribution in Thriller screenplays.

For this reason, another type of dictionary was considered to repeat the same experiment. Instead of using feeling related words, only synonyms suggested in English dictionaries were considered for each analysed emotion. The results of this experiment are shown in Figure 7.

In Figure 7 it is clear that films from the genres Drama and Action have much more evidence of Fear and Sadness related vocabulary than the other genres. Also, Comedy and Romance genres have more words related to the emotion of Joy. These results indicate some pattern in emotion distribution and are potential candidates to be used in a genre classification system. As such, these were used as input for the last model of this system.

## V. GENRE CLASSIFIER

For developing a genre classifier, this approach uses a Multi-layer Perceptron Classifier implementation, because it is a simple approach for a first step into achieving the main goal of this work. The model uses the emotion vocabulary distribution as input and returns a genre classification.

Although the dataset contains a thousand screenplays, it is very unbalanced, and only three genres contained enough examples to be considered as classes for this system: drama

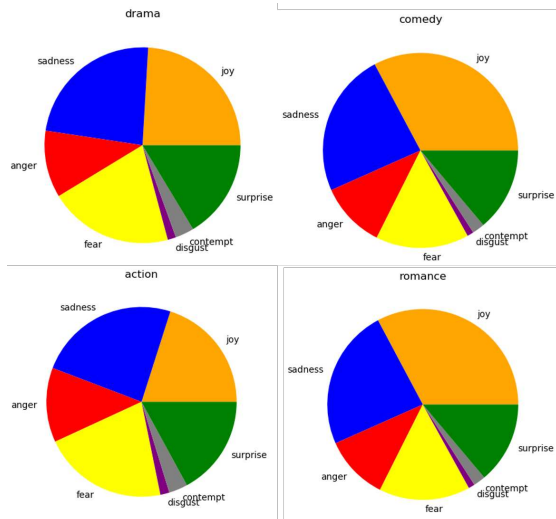


Fig. 7. Simple illustration of synonyms emotion vocabulary distribution in four types of films: Drama, Comedy, Action and Romance

(266 scripts), action (240 scripts) and comedy (239). For this reason, a new genre was created considering romance and family films in the same category, since these genres have very similar emotion vocabulary distributions: predominant vocabulary related to Sadness, Surprise and Fear emotions, inferior presence of vocabulary related to Anger. Thus, the analysis used four genres according to the distribution presented in Figure 8.

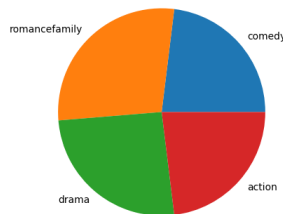


Fig. 8. Dataset distribution with the considered classes for genre classification: Comedy (240 scripts), Romance&Family (294 scripts), Drama (267 scripts) and Action (240 scripts)

In this implementation, only one genre per script was considered. As most scripts are identified to be from more than one genre, only the first named genre was considered, assuming these are listed by order of importance.

Nevertheless, the accuracy obtained in the first steps was approximately 30%. To improve this result, the Action genre was discarded because it was empirically concluded that it was adding noise to the system. This is due to the lack of variations in emotions distribution in the Action genre compared to the remainders, having the other considered genres with more distinct distributions between them. Additionally, the number of variables was reduced to five, discarding the emotions Contempt and Disgust, since these have shown empirically not to play an important role in the description of any of the considered genres, as shown in Figures 4, 5 and 7.

After these improvements, the list of words related to each emotion was also increased, by fetching more related vocabulary using online opensource dictionaries, and the model reached approximately 70% of accuracy rate and a loss lower than 0.1 in the classification between three genres: Drama, comedy and Romance&Family.

## VI. DATA VISUALIZATION

In order to understand thoroughly what this approach is trying to achieve, some data visualization techniques were explored, based on the work Cinemetrics [36]. These are represented in Figure 9, and enable not only the visualization of emotion distribution but also a comparison between the average length of screenplays from each genre.

These coloured semicircle visualizations are graphs where the slices represent each emotion. The colours associated with each emotion were selected based on colour-emotion theories [19] presented in previous sections. The circle's perimeters depend on the average length of the screenplays belonging to each genre. It is noticeable that screenplays from the genre Comedy are usually shorter than screenplays from the genre Thriller. Therefore, this parameter could possibly be considered as a variable for further genre classification.

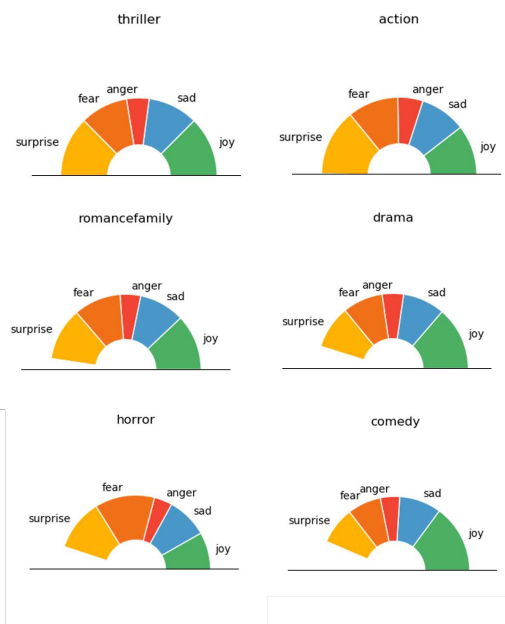


Fig. 9. Representation of screenplays per genre, based on Cinemetrics [36]

## VII. RESULTS

Although the approach on genre classification did not reach a fairly reliable precision, it showed that it is possible to classify the genre of screenplays according to the emotion vocabulary distribution. Consequently, there is a pattern between emotion distribution and genre classification, and this is a naive first approach in this field.

On the other hand, the methods explored for emotion detection did not use any intelligent system, and the dictionaries do not provide sufficient coverage for applications in this field. Nevertheless, the analysis of the data was sufficiently thorough to understand the distribution of emotion-related vocabulary in each analysed genre, where Joy was the most commonly found emotion in Romance/Family and Comedy genres, and Fear and Sadness together the most common emotions found in Drama and Action genres.

## VIII. CONCLUSION

The described approach revealed very promising results in using emotion distribution in screenplays, with natural language processing techniques, for genre classification, with intelligent systems.

To improve the obtained results, it is of utmost importance to obtain emotion-related lists of vocabulary adapted to film screenplay. For this, crowdsourcing techniques should be explored, and the results should require validation from film screenwriters or users in related fields.

Additionally, to improve the second model from the workflow of this approach, other classifiers should be explored in the future, so that other genres can be considered in the classification process.

Lastly, the described theories for basic emotion definition showed to be useful as a starting point, the emotions of Contempt and Disgust were mostly unhelpful for genre classification. As such, the required emotions for this purpose ought to be reviewed as well.

Nevertheless, this first step into automatic classification showed undoubtedly promising results and is a field that will continue to be explored in state-of-the-art approaches.

## REFERENCES

- [1] Z. Rasheed, Y. Sheikh and M. Shah, "On the use of computable features for film classification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 1, pp. 52-64, 2005.
- [2] H. Zhou, T. Hermans, A. V. Karandikar and J. M. Rehg, "Movie genre classification via scene categorization," in *Proceedings of the 18th ACM international conference on Multimedia*, 2010.
- [3] G. S. Simoes, J. Wehrmann, R. C. Barros and D. D. Ruiz, "Movie genre classification with Convolutional Neural Networks," in *2016 International Joint Conference on Neural Networks (IJCNN)*, 2016.
- [4] A. Oliva and A. Torralba, "Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope," *International Journal of Computer Vision*, vol. 42, no. 3, pp. 145-175, 2001.
- [5] J. Wu and J. M. Rehg, "Where am I: Place instance and category recognition using spatial PACT," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 2008.
- [6] D. Arijon, *Grammar of the Film Language*, 1976.
- [7] A. J. Reynertson, "The work of the film director", 1970.
- [8] M. Schilf, Z. Keaton and R. Derek, "The Script's Lab's Encyclopedia of Screen Writing. sl: The Script Lab", 2012, [www.thescriptlab.com/screenwriting/script-tips/657-film-is-visual-show-dont-tell](http://www.thescriptlab.com/screenwriting/script-tips/657-film-is-visual-show-dont-tell). Accessed 3 Jan. 2020.
- [9] Jin, Xin, et al. "Sensitive webpage classification for content advertising." *Proceedings of the 1st international workshop on Data mining and audience intelligence for advertising*. ACM, 2007.
- [10] Terveen, Loren, et al. "PHOAKS: A system for sharing recommendations." *Communications of the ACM* 40.3 (1997): 59-63.
- [11] Alaei, Ali Reza, Susanne Becken, and Bela Stantic. "Sentiment analysis in tourism: capitalizing on big data." *Journal of Travel Research* 58.2 (2019): 175-191.
- [12] Ahmed, Khaled, Neamat El Tazi, and Ahmad Hany Hossny. "Sentiment analysis over social Networks: An overview." *2015 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 2015.
- [13] M. Hesham, B. Hani, N. Fouad and E. Amer, "Smart trailer: Automatic generation of movie trailer using only subtitles," in *2018 First International Workshop on Deep and Representation Learning (IWDRDL)*, 2018.
- [14] Suttles, Jared, and Nancy Ide. "Distant supervision for emotion classification with discrete binary values." *International Conference on Intelligent Text Processing and Computational Linguistics*. Springer, Berlin, Heidelberg, 2013.
- [15] Seyeditabari, Armin, Narges Tabari, and Wlodek Zadrozny. "Emotion Detection in Text: a Review." *arXiv preprint arXiv:1806.00674* (2018).
- [16] The Internet Movie Script Database (IMSDb). [www.imsdb.com](http://www.imsdb.com). Accessed 4 Jan. 2020.
- [17] Plutchik, Robert. "Emotions: A general psychoevolutionary theory." *Approaches to emotion 1984* (1984): 197-219.
- [18] Ekman, Paul. "Basic emotions." *Handbook of cognition and emotion* 98.45-60 (1999): 16.
- [19] Solli, Martin, and Reiner Lenz. "Color based bags-of-emotions." *International Conference on Computer Analysis of Images and Patterns*. Springer, Berlin, Heidelberg, 2009.
- [20] E. S. Tan, "A psychology of the film," *Palgrave Communications*, vol. 4, no. 1, p. 82, 2018.
- [21] T. A. Rashid, "Convolutional Neural Networks based Method for Improving Facial Expression Recognition," *The International Symposium on Intelligent Systems Technologies and Applications*, pp. 73-84, 2016.
- [22] A. Mollahosseini, D. Chan and M. H. Mahoor, "Going deeper in facial expression recognition using deep neural networks," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016.
- [23] A. Dhall, R. Goecke, S. Lucey and T. Gedeon, "Collecting Large, Richly Annotated Facial-Expression Databases from Movies," *IEEE MultiMedia*, vol. 19, no. 3, pp. 34-41, 2012.
- [24] A. Dhall, R. Goecke, S. Lucey and T. D. Gedeon, "Static facial expression analysis in tough conditions: Data, evaluation protocol and benchmark," in *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, 2011.
- [25] I. Kuznetsova, A. Lugmayr and A. Holzinger, "Visualisation Methods of Hierarchical Biological Data: A Survey and Review," *International SERIES on Information Systems and Management in Creative eMedia (CreMedia)*, pp. 32-39, 2018.
- [26] X. Qin, Y. Luo, N. Tang and G. Li, "DeepEye: An automatic big data visualization framework," *Big Data Mining and Analytics*, vol. 1, no. 1, pp. 75-82, 2018.
- [27] N. Bikakis, "Big Data Visualization Tools.," *arXiv preprint arXiv:1801.08336*, 2018.
- [28] L. Wang, G. Wang and C. A. Alexander, "Big Data and Visualization: Methods, Challenges and Technology Progress," *Digital Technologies*, vol. 1, no. 1, pp. 33-38, 2015.
- [29] Agrawal, Rajeev Kadadi, Anirudh Dai, Xiangfeng Andres, Frederic. (2015). *Challenges and Opportunities with Big Data Visualization*. 10.1145/2857218.2857256.
- [30] A who's who guide to the Marvel Cinematic Universe <https://graphics.straitstimes.com/STI/STIMEDIA/Interactives/2018/04/marvel-cinematic-universe-whos-who-interactive/index.html>. Accessed 4 Jan. 2020.
- [31] *Linked Jazz* <https://linkedjazz.org/network/>. Accessed 4 Jan. 2020.
- [32] *Instagram Cities* <http://phototrails.net/instagram-cities/>. Accessed 4 Jan. 2020.
- [33] Tapaswi, Makarand, Martin Bauml, and Rainer Stiefelhagen. "Storygraphs: visualizing character interactions as a timeline." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014.
- [34] Chen, Tao, Aidong Lu, and Shi-Min Hu. "Visual storylines: Semantic visualization of movie sequence." *Computers & Graphics* 36.4 (2012): 241-249.
- [35] Kyung, Hyeonsu, and Jennifer He. "Chromatography: A Quantified Visualization of Movies."
- [36] Brodbeck, Frederic. "Cinematics". *Cinematics.com*, n.d. Web. 9 Mar. 2017 [www.cinematics.lv](http://www.cinematics.lv). Accessed 4 Jan. 2020.
- [37] Willcox, Gloria. "The Feeling Wheel: A tool for expanding awareness of emotions and increasing spontaneity and intimacy." *Transactional Analysis Journal* 12.4 (1982): 274-276.
- [38] K. Robs, "This circular diagram", <https://robbsdramaticlanguages.wordpress.com/2014/07/31/>. Accessed 4 Jan. 2020.

# POSTER SESSION

Chain Fusion: A novel fusion method in multimodal learning

*Joao Barbosa, Bernardo Ramos, Pedro Azevedo, Ricardo Silva and Alejandro Gudino*

Wi-Fi-based network systems design over freshwater: Experimental evaluation using COTS devices

*Miguel Gutiérrez Gaitán, Pedro M. Santos, Luis R. Pinto and Luis Almeida*

# Chain Fusion

A novel fusion method in multimodal learning

João Barbosa, Bernardo Ramos,  
Pedro Azevedo, Ricardo Silva, Alejandro Gudiño



## Abstract

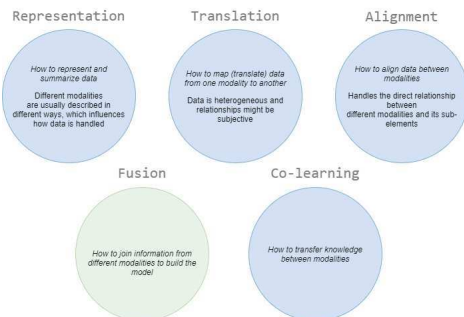
Research in Multimodal Machine Learning has been growing in importance and interest, because it can provide major advantages and results, when compared to unimodal approaches. Methods in this area have reached fields like multimedia classification, audiovisual automatic speech recognition, generation of natural language descriptions of videos, among other applications.

This paper introduces a new method to handle multimodal tasks, to which we call *Chain Fusion*. We provide the main theoretical properties of our method, comparing it to current approaches, such as early and Late Fusion. We also developed an empirical study where we compare unimodal learners, Late Fusion and Chain Fusion. Results show that our method is able to reach performance levels as good or even better than previous methods used in this field.

## Multimodal Machine Learning

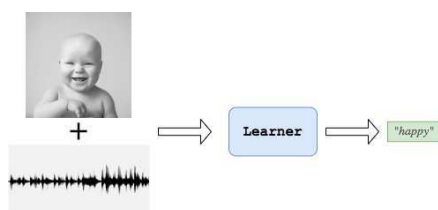
A modality can be referred as the way something is perceived. Multimodal Learning focuses on leveraging data from multiple modalities to build analytic models.

The goal of multimodal learning is to build models that can process and relate information from multiple modalities.



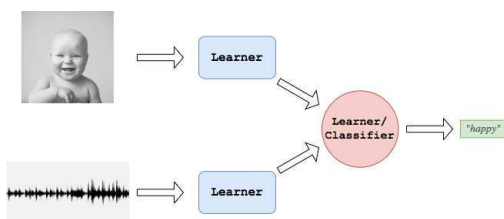
### Early Fusion

In Early Fusion, modalities are joined together right after their features are extracted, often by concatenation before feeding the information to a model learner.



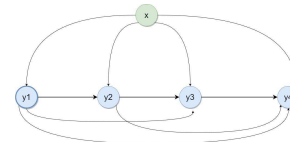
### Late Fusion

Late Fusion integrates the modalities by sending each of them through an individual learner, joining the results of their last layer.

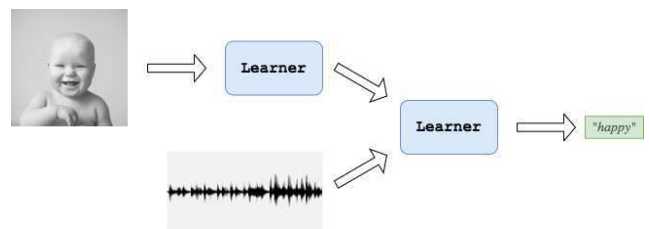


## Chain Fusion

Chain Fusion is inspired in classifier chains - hence our method's name. Classifier chains is a method used in multi-label classification that allows a model to capture inter-dependencies between labels by using predicted targets as attributes.



In Chain Fusion, instead of necessarily using a predicted attribute, we can use the parameters of a model as input to a second one when processing another modality. In a sense, we're using the representation of a modality (parameters of the model's last layer) as complementary information, instead of just how we classify it (the predicted value).



### Comparison of Fusion Methods

	Early	Late	Chain
Uses single model	✓	–	–
Finds inter-modality correlations	✓	–	✓
Modeling flexibility	–	✓	✓
Robust to missing data	–	✓	✓
Parallelizable	✓	✓	–

### Empirical Study

We've implemented a small Chain Fusion project, the dataset *Fashion Product Images* available in Kaggle was used.

Three modalities were used: **Visual** (images of clothing items), **Tabular** (raw data from csv) and **Textual** (a long textual description of the clothing item).

Modalities	Neural Network	Accuracy	Loss	F1 Score
Visual (V)	Pre-trained VGG16	0.6901	1.0252	0.6713
Tabular (Tab)	Two-layered NN	0.6645	0.7264	0.6414
Text (Tex)	Shallow NN	0.8723	0.2480	0.7622
V, Tab	Late Fusion	0.7038	0.7096	0.7003
V, Tab, Tex	Late Fusion	0.7914	0.5660	0.7906
V, Tab	Chain Fusion	0.7146	0.6226	0.7106
V, Tab, Tex	Chain Fusion	0.8043	1.2423	0.8083

### Conclusions

**RQ1:** Is predictive performance improved using Chain Fusion, when compared to unimodal learners?

Chain Fusion improved the F1-Score by about 5% and 4% when joining two and three modalities, respectively.

**RQ2:** How does Chain Fusion compare to Late Fusion?

Chain Fusion slightly outperformed Late Fusion, contrary to our initial beliefs. Our reasoning is that, even though the dataset is composed of sufficient modalities, the textual modality clearly has a higher correlation to the label.

**RQ3:** Does Chain Fusion improve when we add more modalities?

Both the ordering and the number of modalities seem to impact the results. The answer is not yet clear, our results show that the more modalities used the better F1-Score was obtained.

# Wi-Fi-based network systems design over freshwater: Experimental evaluation using COTS devices

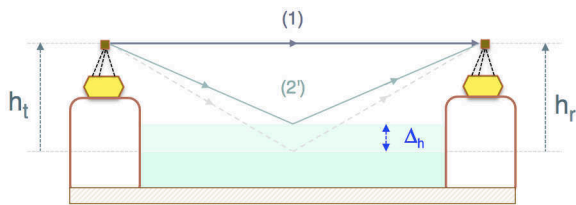
## Motivation

- In the design of **shore-to-shore** and **shore-to-vessel** links, the impact of the **signal reflections** on the surface is often neglected.
- When choosing an antenna height for an inshore node, a typical decision is to use the **largest possible height**; but, this approach can lead to **signal degradation**.

**Objective:** To experimentally assess the impact of surface reflection on the received signal strength of a set of short-and-medium-range shore-to-shore links (<200 m) that use antennas installed at two heights, at a few meters above surface (<3 m).

## Two-ray model

- The **two-ray** is the most fundamental path loss model to account for the influence of **signal reflections** on the received power [1].
- This condition gets further aggravated at near-shore areas as **tides** impose a **variation of the reflection geometry** over time [2].

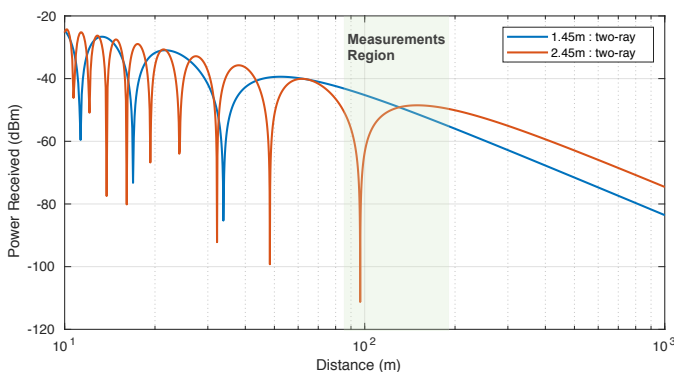


Two-ray model at two different time instants

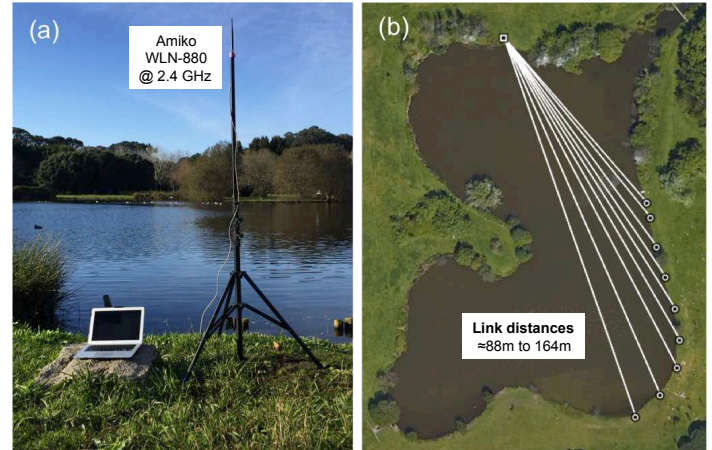
$$P_r = \frac{\lambda^2}{(4\pi d)^2} \left[ 2\sin\left(\frac{2\pi h_t h_r}{\lambda d}\right) \right]^2 P_t G_t G_r$$

## Simulation results

- In the set of link distances and antenna heights that we explore, the **two-ray model** predicts the occurrence of **strong path loss attenuation** (due to the reflected ray) at well-defined distances.

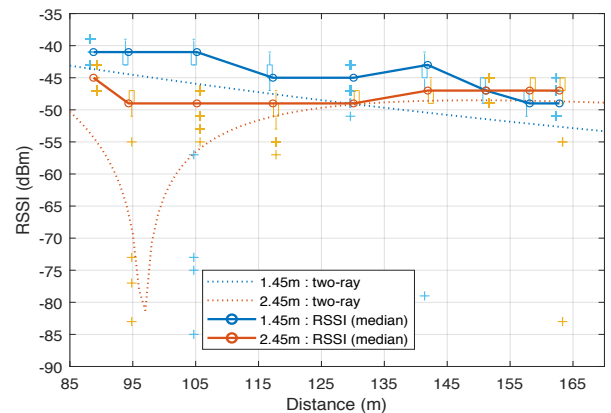


## Testbed



The experimental setup showing: (a) a representative node deployment at the actual location, and (b) the set of links and nodes positions evaluated.

## Experimental Results



Two-ray model (dotted) vs. RSSI measurements at different link distances and antenna heights (boxplot); with median points connected (solid).

## Conclusion & Future work

- We observed **considerable consistency** between packet-based measurements of RSSI and the two-ray model estimates.
- These results provide strength to the claim that both the **two-ray model** and **antenna height adjustment** are key building blocks for effective design of over-water links in coastal environments.

## Reference

- [1] T. Rappaport, Wireless Communications: Principles and Practice. USA: Prentice Hall PTR, 2nd ed., 2002.
- [2] M. Gutiérrez Gaitán, L. Pinto, P. M. Santos, and L. Almeida, "On the two-ray model analysis for overwater links with tidal variations," in 2019 Proceedings of the 11th National Symposium on Informatics (INForum), 2019.



# PAPERS IN ALPHABETICAL ORDER

Affective Computing: Concepts Analysis, Review and Future Directions

Analyzing the Credit-Assignment Problem in Deep Reinforcement Learning

Can Safety-Critical Systems Engineering adopt Agile? Why not?

Chain Fusion: a novel fusion method in multimodal learning

Fault Injection, Detection and Treatment in Simulated Autonomous Vehicles

Finding Emotions in Screenplays for Genre Classification

Live Metrics Visualization: An approach to help software developers

MSDS-OPP: Operator Procedures Prediction in Material Safety Data Sheets

Offensive assessment in social networks

Predicting Predawn Leaf Water Potential up to seven days using Machine Learning

Realistic wheelchair simulator using gazebo and ROS

Teaching a Robot Precision Ball Sports using Deep Reinforcement Learning

Wi-Fi-based network systems design over freshwater: Experimental evaluation using COTS devices

# AUTHORS IN ALPHABETICAL ORDER

Ahmed Fares

Ana Beatriz Cruz, Armando Sousa and Luís Paulo Reis

Carlos Tomás

Daniel Garrido

Eduardo Ribeiro

Eliseu Pereira

Filipa Ivars Silva

Inês Nunes Teixeira

João Barbosa, Bernardo Ramos, Pedro José Azevedo, Ricardo Silva and Alejandro Gudiño

José Aleixo Cruz

Liliana Antão

Miguel Gutiérrez Gaitán, Pedro M. Santos, Luis R. Pinto and Luis Almeida

Sara Fernandes

**U. PORTO**

**DEI** DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

**PRODEI** PROGRAMA DOUTORAL EM ENGENHARIA INFORMÁTICA

**EEFEUP**

**U. PORTO**  
inovação **CGI**

**Primavera Deloitte.**

 IT sector

**Unbabel**



**BOSCH**  
Tecnologia para a vida

**SONAE IM**

**Critical**  
software

**nily.ai**

**Fraunhofer**  
PORTUGAL

**SOGRAPE**  
VINHOS PORTUGAL

PART OF  
**SOGRAPE**  
ORIGINAL  
LEGACY  
WINES

**TAB**

**BLIP.**

A Flutter company