Atas das Oitavas Jornadas de Informática da Universidade de Évora

Évora, 26 de Março de 2018

UNIVERSIDADE DE ÉVORA

UNIVERSIDADE DE ÉVORA
ESCOLA DE CIÊNCIAS E TECNOLOGIA

Departamento de Informática

Título: Atas das Oitavas Jornadas de Informática da Universidade de Évora

Editores: Carlos Pampulim Caldeira

Francisco Coelho

Suporte: Impresso

Formato: Brochado

ISBN: 978-989-8550-57-6

Paper/Author Index

Digital Forensics Research Using Constraint Programming - A preliminary approach

João Calhau, Pedro Salgueiro, Salvador Abreu, and Nuno Goes

Universidade de Évora

Abstract. The world today is becoming more and more digital, as such there is a huge amount of data constantly being created, transmitted and saved every second. Representing this data in digital form, although it brings some advantages it also brings some disadvantages and challenges when we need to make an analysis of the content of said data. For this same reason, it was created the discipline of digital forensics, which focuses on the analysis of digital equipment content. This document introduces an approach on using constraint programming methods in digital forensics analysis, allowing for an easier and more efficient method to analyze digital equipment data.

Keywords: Digital Forensics, Contraint Programming, Security, Declarative Programming

1 Introduction and Motivation

In this work we present a system that makes use of the constraint programming paradigm and methods to solve digital forensics problems, developed in the context of João Calhau MS Thesis. Through the use of constraint programming, we're able to describe a digital forensics problem in a declarative and expressive way, and reach a solution to the problem. The digital forensics problems we're trying to solve are problems such as discovering files that have a specific name, path, type or content and timelines.

The main purpose of the system presented in paper is to allow for an easy and efficient method to search for relevant information in the contents of digital equipment, as previously described. Beyond that, the system must also be able to detect important dates, snapshots, restore points or versions, all these operations mentioned must be performed in a chronological order and without any previous pre-processing of the digital image.

1.1 Constraint Programming

Constraint programming is a powerful paradigm mostly used to solve combinatorial problems. It looks like a simple way to model real world problems but can actually turn into a complex challenge when we want to find solutions for the

problem that is being solved. Constraints can be found in our day to day experiences, almost ubiquitous, representing the conditions that restrict our freedom of decision [15].

In constraint programming, we usually have a set of variables, with an initial domain, to which constraints are applied in order to reduce its domain, and thus reach a solution. Once a constraint is placed on the system it cannot violate another constraint previously applied. This way we can express the requirements of the possible values of the variables [13].

Constraint satisfaction problems are usually solved with the help of solvers. These solvers are essentially search algorithms, usually based on backtracking techniques[11], constraint propagation [12] or local search [6].

1.1.1 Backtracking

Backtracking is a search method that incrementally finds possible candidates to solve the problem. At the same time it removes the candidates that can not be used as a valid solution to the problem [11]. One of the most used examples for this type of search method is the n-queens puzzle, where a set of n queens should be organized, in a $n \times n$ chess board, in such a way that none of the queens can attack each other. Any partial solution that contains two queens that can attack each other is abandoned immediately.

1.1.2 Constraint Propagation

Constraint propagation starts by reducing the variable's domain, strengthening or creating new constraints, reducing the search space, which leads to a problem that is easier to solve. Since this algorithm only reduces the search space reduction of the problem variables, after completion, there is still the need to use another algorithm to solve the problem, which was converted into a simpler problem by the propagators [12].

1.1.3 Local Search

Local search is an incomplete search method to find solutions for a problem. It consists in, iteratively, and with the help of previously defined heuristics, assigning values to the system variables until all the constraints are satisfied. At each step of the iteration, the values of the variables are updated to values *near* the previous value. The algorithm also makes sure of the quantity of constraints it violates so it can have a "cost" associated with the attribution of the values and it can thus tell us if a pre-determined cost has been met [6].

1.2 Digital Forensics

Digital forensics is a very important discipline in criminal investigations where the main device used was an digital device, or to investigates crimes where the evidence may be stored in a digital device. The digital forensics tools have

become a vital appliance to assure we can rebuild information after a cybernetic attack or even if we just want to analyze any type of digital equipment. [8]

It's a complex task to collect evidence/elements in digital equipment, either connected to a criminal activity or not. If that piece of equipment is connected to any type of computer network, with the consequent increase in digital traffic, more difficult it becomes to detect any anomaly or undesirable communication in the network. Thus, the intrusion detection systems have become a very important tool in computer network security. [16]

To collect evidence/data in digital equipment there are many tools capable of analyzing a digital forensics image. Among them, one of the tools that is most used is the *EnCase Forensic* [18], also used in several judicial systems. Besides this tool, which is proprietary and commercial, there are other open source and free of access tools capable of accomplishing the same work. The *Forensic ToolKit (FTK)* [1] and the *Autopsy* [2] are two examples.

2 Known Tools and Other Approaches

In this section we introduce the practical aspects of constraint programming including some libraries and toolkits that are used to model and solve constraint problems. We also describe similar work already done in this area.

2.1 Choco

Choco is a free access and open source library dedicated to constraint programming. It is written in Java and supports several types of variables, including Integers, Booleans, Sets and Reals. It also supports several types of constraints such as AllDifferent and Count, configurable search algorithms and conflict explaining. The first version of Choco was developed in the early 2000s. A few years later, Choco 2 was developed and declared a success in the academic and industrial world. Since then, Choco has been completely re-written and in 2012 the third version of Choco was launched. The current version comes with a simpler API and is denominated Choco 4 [14].

2.2 Gecode

Gecode is a free access, open, portable, accessible and efficient programming environment used to develop systems and applications based on restrictions. Gecode, much like Choco, supports various types of variables and restrictions, among them are Integers, Float and Sets. These variables are used to model problems that are then solved with the help of constraint propagators and search algorithms [17] [9].

2.3 Google OR-Tools

Although Choco and Gecode are two of the most widely used libraries, there are also other new tools, such as the Google OR-Tools. Google Optimization Tools

or OR-Tools is an interface that puts together several linear programming solver and that counts on the use of several types of algorithms such as search algorithms and graph algorithms. What this library has that is so noteworthy is the fact that it doesn't let itself be bound by one language. Although implemented in C++, it is capable of working in other languages like Python, C# or Java.

2.4 The Sleuth Kit

The Sleuth Kit is C library and a collection of tools that allows its users to analyze disc images and restore files from it. The Sleuth Kit is what Autopsy [2], the forensics tool mentioned earlier, uses in it's background jobs. The Sleuth Kit framework allows the user to incorporate additional modules so he can analyze file contents and build automated systems. In addition, the library can be embedded in larger digital forensics tools and command line tools can be used directly to find any kind of proof [3].

Of all the tools The Sleuth Kit has to offer, the most interesting to help us in the type of problem we're trying to solve is the Sorter, which analyzes a file system and organizes what it finds by extension of file. In addition, it provides us details about the organized files, such as the file inode number. The Sorter can also use a separate hash database to ignore files that are known to be good, such as Dynamic-Link Libraries, or dlls, of the windows file system or even know applications.

2.5 Digital Forensics and Constraint Programming

During the analysis and study of the state of the art about constraint programming and digital forensics, no studies were found that combined the two areas. The topic that most resembled this was the use of constraint programming in artificial intelligence in order to make cyber-defense a more reliable and secure method. [19] Considering the reduced number of related works, it is part of the proposed work to see if the use of programming by constraints introduces improvements in terms of processing speed and ease of finding clues or evidence in the cyberspace.

3 Approach

This section introduces our approach for modeling a Digital Forensics Problem as a Constraint Satisfaction Problem using the Choco Solver. We include all data structures needed to model the problem, how they are used to reach a solution to the problem, the methodologies used to analyze and extract the information from the digital evidences, and how the problem is modelled as a CSP.

3.1 Methodology

After acquiring the disk image to be analyzed, it is first processed by the Sorter tool from The Sleuth Kit [3]. The Sorter outputs multiple files with different

names, each name being a pre-determined type of file, such as archive, executable or data. Each output file contains all the information the Sorter collects such as the file path in the file system, the file type, the image name (from where the data was extracted) and the inode number, which is an internal representation of that particular file in the file system. With this information there isn't much we can do, because the information hasn't been processed yet, this means we need to store it in some kind of data structure. When thinking about what type of data structures to use we figured out that having the data organized in various different ways would make it easier to extract the information in a later date, so, instead of one data structure we decided on using three different data structures. These data structures are described in Section 3.1.1

To populate these data structures we parse the data of files created by the Sorter tool. That can be easily achieved by reading the contents of said files and storing them in some kind of "wrapper" in the data structures. This wrapper ended up being a representation of the inodes containing the inode number, the file path, the file type and the file name. The files outputted from the sorter always have the same type of structure, three lines of text followed by and empty line, this can be seen as an example in figure 2. Also the three lines of text always come in the same format, first line is file path and name, second line is file format and third line is image name and inode number. All we have to do is iterate over those lines four at a time, gather the information, build the inode representation and store it on the data structures.

The whole process of extracting the data from the file system can be shortened into the small flow diagram seen in figure 1.

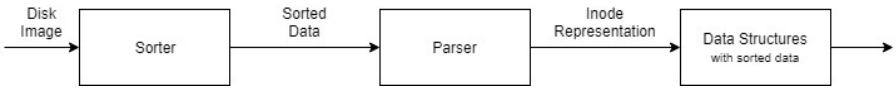


Fig. 1. Flow diagram

```
1  idle_master/CSteamworks.dll
2    PE32 executable (DLL) (GUI) Intel 80386, for MS Windows
3    Image: pen_4_dd.dd  Inode: 40-128-1
4
5  idle_master/HtmlAgilityPack.dll
6    PE32 executable (DLL) (console) Intel 80386 Mono/.Net assembly, for MS Windows
7    Image: pen_4_dd.dd  Inode: 42-128-1
8
9  idle_master/IdleMaster.exe
10   PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
11   Image: pen_4_dd.dd  Inode: 43-128-1
```

Fig. 2. Example of sorter output for executable files

3.1.1 Data Structures

As said before, we decided to go for three different data structures, the first one would be used to store the representation of the inodes, these didn't need to be stored in any particular order so we went for a data structure that didn't have order, that was easy and efficient to use, an Hash Map of inodes. In the second structure we decided to store the inode representations, but this time by path, that is, we use the same type of structure as the first one, but with a twist, this time we would use a Hash Map of Linked Lists of inodes because they are going to be organized without any particular order, but more than one inode can have the same path, this means we would need to insert various inodes with the same key in the Hash Map, and that is just not possible, unless we insert the inode in the linked list at that key position. Finally, the third data structure would be used to store the inodes by type and as the file types are always the same (because the sorter always outputs the same type of file types) we decided on using a simple array (of fixed position) of Linked Lists, the array is always the same, what changes is the Linked Lists inside said array.

3.2 Modelling

As previously mentioned, the files extracted from the file system are sorted into various different types and have various types of data extracted from them, such as type, path, name and inode number. Because each file has a different inode number, which is an integer, we decided to use the inodes to represent our files in our Constraint Satisfaction Problem.

The framework we ended up deciding on using was Choco, due to it's good rating, good documentation and language familiarity (Java, when opposed to C++). We now had our variable's domain, all the inode number in the file system, but we still did not know which type of variable we would use. Choco has four different types of variables available with which we could model our problem with: Integers (IntVar), Booleans (BoolVar), Sets (SetVar) and Reals (RealVar). We wound up deciding on Integer Set Variables, or SetVars, because the final solution of our solver would need to be a set of one or more integers.

In Choco Solver, SetVars are defined by a domain that is composed of two separate domains, the LB and the UB, these being Lower Bound and Upper Bound, respectively. The Lower Bound is a set of integers that must belong to every solution and the Upper Bound is composed of the set of integers that may be part of the final solution [5]. In our case, when creating the variable with which we are going to work with, the Lower Bound will be left empty, because we do not know what files we want yet, and the Upper Bound will be composed of all the inodes existing in the first data structure (the Inode Data Structure). Finally all that is left to do is create our custom constraints and apply them to our variable so we can restrict it's domain.

3.3 Constraints and Propagators

To implement a new constraint in Choco solver, first we need to create a propagator. A propagator declares a filtering algorithm that can be applied to the Variables that model the problem, in order to reduce their domain [4]. Since this work is still in it's early stages we decided to create a couple of simple propagators to test the validity of our approach. We decided to implement the following propagators: 1) file type propagator that restricts our domain according to the given type of file passed as argument, it restricts the domain based on the type data structure created before; 2) file path propagator that restricts our domain according to the path passed as argument, it restricts the domain based on the path data structure created before. Both propagators were build to work with the variables used to model the problem, SetVars.

For the propagators to work, we have to implement the following methods methods: `propagate` and `isEntailed`. The method `propagate` is pretty straight forward, it should restrict the domain according to what we need. The method `isEntailed` is straight forward as well, all we need to do here is tell the propagator when the problem has a solution or not, or if it is simply undetermined. These methods are described in detail in Listings 1 and 2.

Listing 1 propagate method

```
for each value in UB do
    if value is not in corresponding structure then
        Remove value from UB
    end if
end for
```

Listing 2 isEntailed method

```
if UB is empty then
    Problem is impossible to solve
else
    Problem has possible solution
end if
```

Both propagators work in a similar way, they take the Upper Bound of the SetVar, iterate over it and remove any inode that is not in the desirable data structure. For example, if we want to propagate an executable type, our SetVar is composed of two domains: and empty one $\{\}$ (the Lower Bound) and one with three inodes $\{100, 101, 102\}$ (the Upper Bound). Our type structure only has one inode in the executable division $\{100\}$, what the propagator would do in this situation is remove every inode that is not in the type structure from the SetVar which would leave it like so: $SetVar = \{\}, \{100\}$. Of course this is

only an example, and on top of this another constraint could be applied, like a path constraint (never another type constraint, because that would just leave the Upper Bound domain empty every time).

3.4 Experimental results

In our experimental phase we decided to start by analyzing something small, like a disc or a pen drive, what ended up being chosen was the latter. We took a simple four gigabyte pen drive and passed it through FTK Imager [1] to obtain it's image. After obtaining the image we wanted, we passed it through the Sorter and waited for the contents of the pen drive to be sorted. After the Sorter finished it's work we would have to choose the constraints we would want to use later on.

For a first test we chose only one constraint, a type constraint that would restrict the domain to only the files that had an archive type. We booted the program and, sure enough, the resulting solution outputted only the inodes belonging to the files that had an archive type as can be seen in figure 3. Seeing as the first test went well, we decided to apply a second constraint on top of the first one, this time we chose the unknown type for the type constraint and a specific path in the file system "LVOC/LVOC/", this way the program should output only seven file inodes and it did as observed in figure 4.

```
D:\Java\jdk-8\bin\java ...
Inodes found:
Inode(45, idle_master.zip, idle_master, Archive)
Inode(49, LVOC.zip, LVOC, Archive)
Inode(717, Exclusive Collection of E3 2016 Cards.zip, ubi 30, Archive)
Inode(718, Exclusive Digital Posters from E3 2016.zip, ubi 30, Archive)
Inode(719, For Honor GIFs.zip, ubi 30, Archive)
Inode(720, Ghost Recon Wildlands GIFs.zip, ubi 30, Archive)
Inode(721, Holiday Wallpaper.zip, ubi 30, Archive)
Inode(722, Just Dance Greeting Card.zip, ubi 30, Archive)
Inode(723, Rabbids Holiday Goodies.zip, ubi 30, Archive)
Inode(724, Rayman GIF.zip, ubi 30, Archive)
Inode(725, Ubi30 360 Image.zip, ubi 30, Archive)
Inode(726, Ubi30 Exclusive GIF.zip, ubi 30, Archive)
Inode(727, Ubisoft Cocktail Recipes.zip, ubi 30, Archive)
Inode(728, Ubisoft Dessert Recipes.zip, ubi 30, Archive)
Inode(729, Ubisoft DIY Advent Calendar.zip, ubi 30, Archive)
Inode(730, Ubisoft Gift Tags.zip, ubi 30, Archive)
Inode(731, Ubisoft Wrapping Paper.zip, ubi 30, Archive)
Inode(732, Wallpaper for Mobile.zip, ubi 30, Archive)
Inode(733, Watch_Dogs 2 Wallpaper.zip, ubi 30, Archive)
Inode(734, Werewolves Within Wallpaper.zip, ubi 30, Archive)
```

Fig. 3. Program output for the first test

```
D:\Java\jdk-8\bin\java ...
Inodes found:
Inode(40, CSteamworks.dll, idle_master, Exec)
Inode(42, HtmlAgilityPack.dll, idle_master, Exec)
Inode(43, IdleMaster.exe, idle_master, Exec)
Inode(46, Newtonsoft.Json.dll, idle_master, Exec)
Inode(47, steam-idle.exe, idle_master, Exec)
Inode(50, Steamworks.NET.dll, idle_master, Exec)
Inode(51, steam_api.dll, idle_master, Exec)
```

Fig. 4. Program output for the second test

4 Conclusion and Future Work

From the experimental phase, we can conclude that what took the most time to finish was the extraction of the image with the help of FTK Imager [1], this took about five minutes for a four gigabyte pen drive that had about two gigabytes of data. The Sorter ran in about twenty-five seconds while the Java program ran in under one second. We can also conclude that the program works and restricts the domain as it should, in a very short amount of time, with both devised constraints placed at the same time. What we would need to do in terms of future work, would be the creation of more constraints, like restricting the domain further to files that only have certain keywords or files that have been altered recently or in a certain period of time.

References

1. AccessData. Forensic toolkit, 2017.
2. Brian Carrier. Autopsy - the sleuth kit, 2017.
3. Brian Carrier. The sleuth kit, 2017.
4. Choco-Solver. Class propagator api, 2018.
5. Choco-Solver. Interface setvar api, 2018.
6. Rina Dechter. *Constraint Processing*. 2003.
7. Edward Delp, Nasir Memon, and Min Wu. Digital forensics [From the Guest Editors]. *IEEE Signal Processing Magazine*, 26(2):14–15, 2009.
8. Simson L. Garfinkel. Digital forensics research: The next 10 years. *Digital Investigation*, 7(SUPPL.), 2010.
9. Gecode Team. Gecode: Generic constraint development environment, 2006.
10. Google. Google optimization tools, 2017.
11. Donald E. Knuth. The Art of Computer Programming, Vol. 1: Fundamental Algorithms, 1997.
12. Christophe Lecoutre. *Constraint Networks: Techniques and Algorithms*. 2010.
13. Justin Pearson and Peter Jeavons. A Survey of Tractable Constraint Satisfaction Problems. pages 1–42, 1997.

14. Charles Prud'homme, Jean-Guillaume Fages, and Xavier Lorca. *Choco Solver Documentation*. TASC, INRIA Rennes, LINA CNRS UMR 6241, COSLING S.A.S., 2016.
15. F. Rossi, P. Van Beek, and T. Walsh. Handbook of Constraint Programming (Foundations of Artificial Intelligence). pages 281–322, 2006.
16. Pedro Salgueiro, Daniel Diaz, Isabel Brito, and Salvador Abreu. Using constraints for intrusion detection: The NeMODe system. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6539 LNCS:115–129, 2011.
17. Christian Schulte, Guido Tack, and Mikael Z. Lagerkvist. Modeling. In Christian Schulte, Guido Tack, and Mikael Z. Lagerkvist, editors, *Modeling and Programming with Gecode*. 2017. Corresponds to Gecode 5.1.0.
18. Guidance Software. Encase forensic, 2017.
19. Enn Tyugu. Artificial intelligence in cyber defense. *2011 3rd International Conference on Cyber Conflict*, pages 1–11, 2011.

Secure Framework for Data Sharing on Clouds Based on Blockchain

André Figueira and Pedro Salgueiro

Universidade de Évora

Abstract. Blockchain is a relatively new and disruptive technology that is considered a distributed database working as a ledger, having the ability to facilitate the recording of transactions and tracking of assets. Sharing data through third party services using unknown methods is a delicate process regarding the privacy and security aspects. These two aspects are crucial points when it comes to personal and private data. This paper presents an initial version of how the Blockchain and its implementations like Hyperledger Fabric or Ethereum Project can be used to provide a safe and secure data sharing mechanism and a method to keep control over the rules and policies that enable data access.

Keywords: Blockchain, Data Sharing, Smart Contracts, Security, Clouds.

1 Introduction

Data sharing on cloud platforms is very common and secure in modern times, but sharing on cloud platforms and relying on third parties to manage and maintain data secure and private with unknown means of doing it, is not an easy task. It requires a degree of trust that most are not comfortable with, especially when it means to trust others with private, sensitive or confidential information. This leads to issues like who can access the data, who has accessed the data, for how long can data be accessed, who can alter the data, etc... These are problems associated with trust, authorization, ownership of data that concern users especially on a corporate level. By various approaches of storage with the addition of the Blockchain and smart contracts it can be a viable solution to the problem, and by taking advantage of platforms like Hyperledger Fabric or the Ethereum Project which already have proved concepts and applications as well as a stable implementation of blockchain and smart contracts, can be a great addition to the proposed work, allowing the implementation of a secure data sharing mechanism and methods to control the rules and policies of data access in the possible way.

This paper describes the work of André Figueira in the context of his MS thesis, Secure Framework for Data Sharing on Clouds based on Blockchain and therefore will introduce the topics approached by the thesis itself. These are Blockchain, Smart Contracts, Blockchain Implementations, approaches to the problem, already existing similar services or related work and finally an initial

implementation. The thesis topic is highly centered on the security and privacy aspects of data shared through cloud based service, aspects that are crucial when it comes to personal, private or confidential data which is of great concern, when dealing with small, individual users up to corporations who have to trust others to make sure these aspects are verified while not truly owning their data. The topic of the thesis revolves around the study and implementation of a cloud based framework for data sharing based on Blockchain, a disruptive and relatively new technology. The capabilities and properties of Blockchain may prove to be a strong and viable solution for the privacy, safety and security of data and most importantly allow for a true sense of ownership while also providing no single point of failure.

2 State of the Art

The proposed work is highly centered on the security of data sharing and how to best achieve it. The assessment of the state of the art will address the concept of the blockchain technology and additionally the concepts of smart contracts. It will also consider existing blockchain implementations, able to create applications on top of them, which demonstrates that its capabilities can be advantageous for the proposed objectives. Besides that, some related work is described to better understand how the described existing services handle similar issues, in order how to best understand and implement the proposed system.

2.1 Blockchain

Blockchain is considered a distributed database working as a ledger, "*having the ability to facilitate the recording of transactions and tracking of assets.*" [7]. It consists of individual blocks containing information of transactions of some sort, chained together by a digital signature, hash link or any other appropriate method, where a block references the previous block resulting in a chain of blocks, thus named a blockchain. The Blockchain holds information of tangible or intangible assets having the capability to reduce risks, increase visibility, faster automated processes and networking costs in the process as opposed to other commonly used methods [7]. Often described as decentralized, it is replicated across multiple nodes in a network, thus every node maintains the same copy of the blockchain, although in some networks, nodes may be prevented from holding a copy of the blockchain (private networks). The blockchain, since replicated to every node makes it persistent to change, immutable, unable to be corrupted. As mentioned the Blockchain is made by blocks, each block is a set of transactions of some sort that were checked and accepted by the network rules of consensus. Once accepted, every node that holds the blockchain adds the block to their copy of blockchain, which every node has an equal copy of. This together with the chain that links blocks prevents the blockchain from being modified and any attempt to modify is easily detected. In other words, the block and the transactions that are part of are no longer reversible and the change in the

blockchain is final, the effect is permanent. For example, a SQL database when an INSERT is performed the line can be easily changed or removed, but in this case there can be no UPDATE or DELETE on that line or any other, never. As it is replicated across multiple nodes, the blockchain is always available, having no single point of failure because every node can answer as of being part of a peer to peer network. In terms of blockchain network, it can be public or private, depending on the situation [10].

- Public Blockchain Network, any user can write and read data, interact with the blockchain and anyone can join the network, an example is the Bitcoin network.
- Private Blockchain Network, the users are assumed to be known and trusted but there are enforced restrictions to participants who can write and read data, maintain the blockchain, etc...

The Blockchain was first implemented and conceptualized by *Satoshi Nakamoto*, a name used by an unknown person or group, to be used in Bitcoin, being its core component, also created by Satoshi Nakamoto. The blockchain was created essentially to be an immutable ledger to record transactions, "*becoming a chain of digital signatures that defines an electronic coin*" [11] , making it possible to create a decentralized digital currency (Bitcoin) allowing "*online payments to be sent directly from one party to another without going through a financial institution*" [11]. Becoming the key aspect in solving the problem of "double spending" [11], which is the idea of using the same amount more than once, a problem that is often associated with digital currencies.

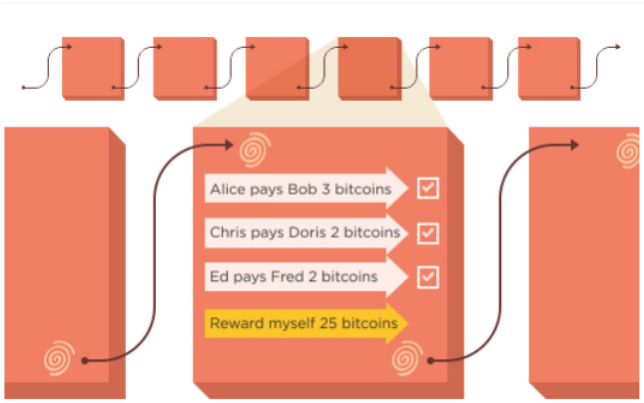


Fig. 1. A bitcoin blockchain block [10].

Figure 1, by Antony Lewis *A Gentle Introduction to Blockchain Technology* (p. 8) [10], shows the contents of a bitcoin blockchain block, it contains information from various transactions which have been confirmed by the network, the

reward for the node/nodes who successfully contributed in creating the block and a digital signature which links blocks together.

2.2 Smart Contracts

Smart contracts, often compared to vending machines, are applications, written in a programming language (Golang, Java, ...) by users of the network, "*that run exactly as programmed without any possibility of downtime, censorship, fraud or third party interference*" [1]. A smart contract has rules, conditions and penalties written in code, and when invoked, automatically determines what is the appropriate action according to the set of rules and conditions written in the contract for that given situation, thus enforcing, validating and verifying and with the support of the blockchain removes the need to rely on middleman to confirm the conditions and others processes of a contract, or having the fear of third party malicious actions interfering, as these contracts are automatically invoked when a specified situation is triggered. These contracts are stored in the blockchain and as such all data is stored within the blockchain and is immutable, increasing safety, visibility and allowing to minimize trust by reducing human judgments that could influence the transactions.

2.3 Blockchain Implementations

The blockchain concept, since the Bitcoin, inspired the creation of multiple platforms with their own custom built blockchains. These implementations have their own objectives, serving different needs. The Ethereum Project and Hyperledger Fabric are two of the blockchain implementations which are the basis of various systems and solutions and whose capabilities can greatly benefit the topics approached in this work. In addition these systems are highly complex and highly technical, understanding and taking maximum advantage of their capabilities is not a simple task, it requires a great amount of effort and time in order to understand how to best use it and how to best achieve the goals previously stated.

Ethereum Project by the Ethereum Foundation, has a custom built blockchain with a smart contracts implementation that is used by its users to create blockchain applications to be run on their public blockchain. This enables to have "*an enormously powerful shared global infrastructure that can move value around and represent the ownership of property*" [1].

Ethereum reaches consensus by proof of work, or known by another term, mining, which is a common word in the digital currency world. This refers to the calculation needed to determine the legitimacy of transactions via mathematical proof. Clients requesting operations to be executed by the blockchain network must reward the nodes (named *miners*) who execute the operations, this is in the form of Ether, the platform own digital currency, making it a fee based platform. Since it is a public network, every node holds a copy of the blockchain,

and clients who interact with the blockchain must connect via a node. Every node can mine (miners), but it is entirely the choice of the node to do so.

The Blockchain in Ethereum is considered a *"cryptographically secure transactional singleton machine with shared-state."* [13], essentially a "*transaction-based state machine that will read a series of inputs and, based on those inputs, will transition to a new state*" [9]. For a state transition to occur, as stated before, must be checked via proof of work where when enough transactions are checked they are bundled together and a block is created and added to the blockchain. The Ethereum Project blockchain is similar to the Bitcoin blockchain but "*Ethereum blocks contain a copy of both the transaction list and the most recent state*" [5] of the transactions that took place. Smart contracts in Ethereum "*are account holding objects on the ethereum blockchain. They contain code functions and can interact with other contracts, make decisions, store data, and send ether to others*" [1].

Hyperledger Fabric by the Linux Foundation, is a platform for a distributed system of records, or a distributed ledger, allowing application with high confidentiality, scalability, layers of consensus and security using its own blockchain implementation and own version of smart contracts (named chaincode). The Hyperledger Fabric allows private networks and "sub private" networks (named channels) of decentralized nodes highly focused on consensus, but there is no proof of work. Consensus in this platform is only achieved in its entirety when certain policy criteria are checked [8]. A transaction in order to take place must be endorsed by certain nodes, according to certain endorsement policies. After that, there should exist consensus among the nodes who are allowed to update the blockchain. Thus, a very complex system of consensus and understanding among the nodes must take place when either invoking or deploying a transaction. In the network there are two kinds of nodes, validating nodes which handles the verification of the transactions interacting with the blockchain and so its maintenance. And non-validating nodes which act as a bridge ensuring communication between clients and validating peers, these do not interact with the blockchain but can verify transactions able to endorse them under certain policies. [6] In contrary to Ethereum Project, Hyperledger Fabric does not rely on cryptocurrency in order to work (no need for "*mining*").

The Blockchain being the most important aspect of the Hyperledger Fabric, its purpose is holding state and ledger data, run chaincode and execute transactions [8]. Chaincode is Hyperledger Fabric version of smart contracts that are kept on the blockchain handling the application and business logic (agreed by the network) and waiting to be invoked by someone. "*Chaincode initializes and manages ledger state through transactions submitted by applications*" [8], also known as blockchain applications. Transactions that undergoes in the blockchain are of two kinds, deploy and invoke. Deploy transactions create chaincode and when successful it is installed on blockchain where it resides, unable to be changed, and waiting to be invoked. Invoked transactions must be endorsed and when allowed and successful, permits the client to execute functions from a previously deployed

chaincode. Regarding the blockchain, State and Ledger are datastructure components of the blockchain. State is the latest state of transactions, working in key/value pairs, it reflects only successful state transitions of transactions, this means, it contains the latest values for any given key, and can always be reconstructed via the Ledger. Ledger, as the names suggests, is a system of records, it contains all the history of the state transitions of all transactions that took place, both invalid and succeeded, thus keeping a record of everything that undergoes in the network. The ledger is a definite source of data and essentially being the blockchain as the State component can be constructed via the Ledger. In short, *"the ledger is constructed by the ordering service as a totally ordered hashchain of blocks of (valid or invalid) transactions"* [8].

2.4 Possible Approaches

There are multiple approaches in order to achieve the stated goals, each with their own challenges and differences. From fully decentralized to decentralized approaches but the most important point is that all of these maintain the blockchain as its most crucial and core component. Essentially the blockchain provides a record of the files that exist in the network, keeping record of the share permissions, owner of the file, availability of the file, etc... (stated in Section 3) but not the file itself, as that could generate complications when dealing with files of varying sizes and files that would require constant update, which would generate various uneven blocks and could comprise the efficiency of the blockchain. This record allows to locate the file, present the users their files and shared files and help retrieve the original file from its corresponding repository (either centralized or distributed) while still ensuring complete and total ownership. (as it is one of the fundamental aspects of the blockchain). Smart contracts would be used, as the only way, to allow users to interact with the blockchain in other to perform various operations, like register a file in the blockchain.

Distributed System With this approach, a file is to be fragmented into smaller equal encrypted pieces and distributed to the network to be kept in other peer's physical machines, according to rules stated in the smart contract. The blockchain would then handle all the relevant information in order to log, locate, decrypt and rebuild the original file while still providing complete ownership of the file.

Client and Storage Provider This approach requires a trust connection (since it is a private network it is assumed) between two peers where a client submits a request to store files into the another peer's file system according to the rules stated in the smart contract. The blockchain would handle the records of the files, location and other relevant information in order to retrieve and decrypt files while still providing complete ownership of the files.

Hybrid System This approach would require a centralized approach meaning a central authority where files would be kept in a central repository while using the blockchain (as the decentralized component of the network) for support in terms of ownership and other relevant information to log, retrieve and decrypt user's files when requested. Files would be shared according to rules stated on smart contracts. This approach would mean have a centralized component (the repository) and a decentralized component (Blockchain).

2.5 Related Work

The blockchain concept led to the rise of blockchain based services. Cloud backend services are obviously among them. Storj and Sia are two of these services which reflect topics approached by this thesis. Any of the two presented solutions imply costs lower than more known and traditional methods like Google Drive, Dropbox, Amazon S3, for example. Although they use a relatively new innovative and disruptive technology, they provide far more affordable decentralized solutions to cloud storage due to the nature of the Blockchain, but nonetheless have their flaws.

Storj is an open source but not entirely fully decentralized backend cloud storage service implemented on top of the Ethereum Blockchain that allows its participants to rent unused hard drive using the public blockchain as a ledger to keep track of shared files. It does require trust on third parties (named "*bridges*") to connect users to storage participants in order to pay/storage. Each block in the blockchain contains hash functions, keys, file locations, etc.... Shared files are fragmented and spread across the decentralized network and encrypted, guaranteeing the owner, he is the only one who has access to the complete file. It uses its own digital currency to reward the participants for keeping the network up and running although allows payments to be done in other currencies besides the networks [4].

But consequently has issues, has it is not fully decentralized meaning it has single point of failure. Making it vulnerable to Distributed Denial of Service due to the "bridges" incapacitating the entire network.

Sia is an open source project, providing a fully decentralized with no single point of failure cloud backend service where participants of the network can rent his unused hard drive space and host files. *"Instead of renting storage from a centralized provider, peers on Sia rent storage from each other. A blockchain, similar to Bitcoin, is used for this purpose"* [12] . It works by creating a contract between the storage provider and client's data and periodically submit proof (verifiable through the Blockchain) of their continued storage until the contract expires. While redundantly storing data across multiple hosts in order to achieve high availability. The platform uses its own digital currency in order to reward the participants of the network [3].

But like Storj, it has is issues. In order to "help" the network it requires synchronizing with the entire blockchain which can take a considerable amount of time due to the size of the blockchain and also having some scalability issues in handling large volumes of data due to limitations of the blockchain. In addition, payments are only done in the network currency Siacoin, that could be viewed as a downside.

3 Initial Version

In this section, we present a first approach on how to implement a mechanism to share files using the blockchain technology, in order to understand how components interact with each other allowing to improve the implementation in the future. We describe how the blockchain is used to share files, including what is stored in the blockchain, the contents of the smart contract and an application that allows users to register themselves on to the network and interact with the blockchain and the storage server. The Blockchain implementation being used is Hyperledger Fabric.

In this first version a centralized repository will be used to store the files of the users while the Blockchain and Smart Contracts provides support.

As for the application, currently written in Node.js, it is divided into two parts. The first part allows connection to peers of the Hyperledger Fabric network thus enabling interaction with the blockchain via smart contracts (stated in Section 3.2) and the second part, interaction with the central repository (not necessarily through the use of smart contracts). The purpose of this user application is to display a series of tools enabling the interaction with the central repository and the blockchain simultaneously. A user is able to see shared and owned files, store, delete, share and check history of files via the application by simultaneously calling appropriate smart contract functions (which will be stated in Section 3.1) in order to provide coordination between the blockchain and the central repository, so it can be verified that the users have access to what they are supposed to have access to.

An example would be the application displaying the owned and/or shared files by requesting this information from the blockchain and if the user wanted to download a shared file, then this application would, again check the blockchain for permissions and then retrieve from the repository.

3.1 Smart Contract for File Management

This Smart Contract contains several functions, written in Golang. Describes how the users of the network can interact with the blockchain, thus defining a set of transaction instructions to manage the files. As previously stated, transactions allow to modify a corresponding state that allows the management of each individual file in the network. It uses the Golang shim package, which *"provides APIs for the chaincode to access its state variables, transaction context and call other chaincodes"* [2].

For this version, file management for each file is structured by a total of six fields, which are stored in the blockchain. The values of these can be modified by invoke transactions, that correspond to functions of the smart contract. These six fields are: File Identifier, File Location, File Owner, File Share Permissions, Timestamp and File Status with more to be added in later versions, noting that these existing ones can change. These fields are identified by a single unique key, currently a sequential identifier. This is necessary as it enables to identify the file within the blockchain with increased ease. In future version this key can be modified to include important relevant information about the transaction, thus working in a key/value pairs.

The following describes in more detail these fields:

- File Identifier, this field refers to the identity of a file so it can be associated to the owner.
- File Location, this field is essential for file recovery which refers to the location of the file or locations of the fragments of the file (as it was stated in Section 2.4).
- File Owner, this field refers to the identity of the owner of the file, which binds the file to the owner giving access to different functions stated in Section 3.2.
- Permissions, this field refers to who has permissions to access the file (registered users), which can be given and removed only by the owner of the file. This field will most likely go through a process of change.
- Timestamp, this field refers to the timestamp of at the time of an invoke. This field is of great importance as of being able, for example, to determine when a permission was given and to who was granted, thus being able to log file sharing.
- File Status, this field refers to the status of the file (available, not available, deleted, etc. . .).

In this initial version the File Location is not used, as we are using a central repository as previously mentioned. In the near future it may prove useful as the repository may shift from centralized to distributed, as stated in Section 2.4.

The functions offered by this smart contract are listed below. These smart contract functions are the ones that can interact with the blockchain.

- Init Ledger, this function is the first that is usually called, simply instantiating the smart contract on the channel it has been installed on. Currently it is only initializing the chaincode docker containers for the target peers.
- Register File, this function registers a new file in the blockchain, with the six fields described in Section 3.1, by default it has no permissions (permissions field is empty) at the time it is invoked.
- Query File, this function is self explanatory. Currently it uses the unique key as a function argument, the information however can only be seen by the owner of the file. Later versions may have additional fields (even already existing ones) which can be viewed by any participant in order to have some external verification of the file sharing that has occurred overtime.

– Query Files, this function takes a flag, which determines if it should return only shared files, only owned files, or both, given that flag it scans the blockchain and returns one of the options that was previously stated.
– Query Creator Files, a functions with no arguments, that returns all the information of each file of the owner, which is the six fields plus the unique key, and only the owner can see his complete list of files.
– Update File Status, function that updates the status of a file, from available, unavailable, deleted, etc... This is can only be done by the owner of the file by updating the value of the status field of a transaction.
– Update File Permissions, same as before but for sharing the file with other users by updating the permissions field, only the owner can remove or give permissions, with no exception. In the following iterations, the issue of file permissions must be more evolved and robust so a very possible scenario is the creation of another Smart Contract that only handles the creation and handling of "file sharing contracts" between users to impose restrictions on file sharing and handle all the logic around it as well.
– Check History of File, this function is self explanatory. Using the unique key that identifies a file (with its six fields) returns the history of changes to the six fields stated previously.

3.2 Users Apps

Applications, currently written in Node.js, it relies in fabric node.js sdk that allows users to connect to peers of the network (which can also be users) enabling them to interact with the blockchain. This however, is only true if registered and accepted as a user of the network, which is achieved by communicating with the Certificate Authority within the network that generates certificates. Interaction with the blockchain is only allowed through the use of smart contracts, thus being only able to query and update the ledger and getting responses from it. Besides interaction with the blockchain it connects users to the storage server allowing coordination between the two components.

An example would be the application displaying the owned and/or shared files by requesting this information from the blockchain via smart contract and if the user wanted to download a shared file, then this application would firstly use a smart contract function to determine if the user still has access to that specific file and if true would retrieve it from the repository, if not would return an error message.

3.3 Blockchain Interaction Diagram

Fig. 2 illustrates how an application interacts with the blockchain, as it was stated previously, it can only do so via smart contracts[1]. The user application makes a request, this only achieved by the use of smart contracts, the smart

[1] Fig. 2 was based on the diagram presented in hyperledger-fabricdocs (Chap. 9, p. 50) [8]

contract executes one of its functions which can fulfill the request and returns the user the result from the blockchain, this assuming of course the user is registered in the network, and has access to utilize the smart contract and the request had the correct structure.

The ledger latest state or the State component, which as it was stated in Section 2.3, is key/value pair, which contains the most recent values for any given key and is constructed using the ledger.

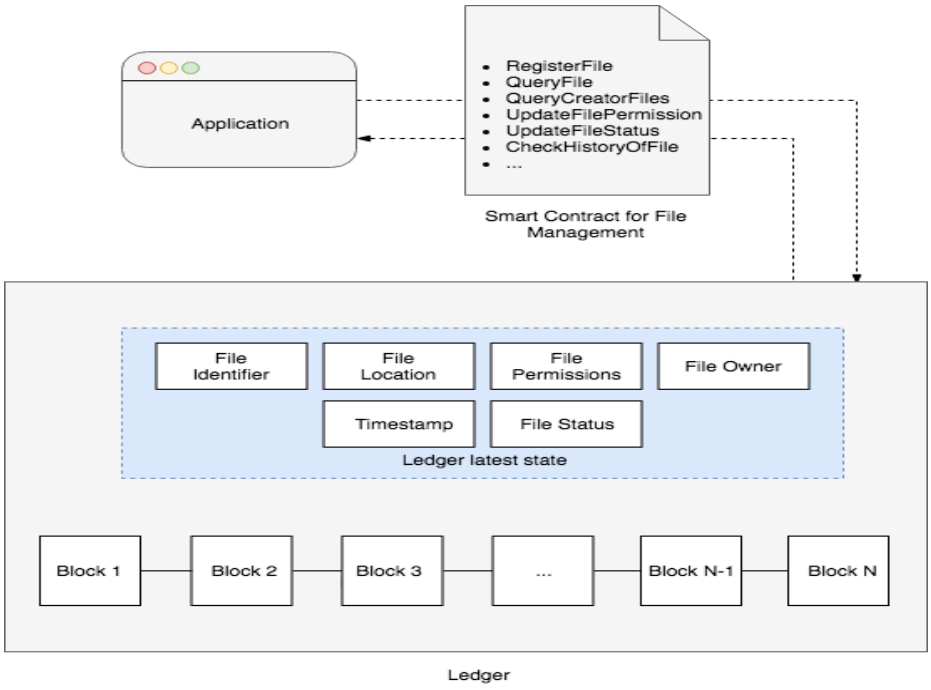


Fig. 2. Blockchain Interaction Diagram

4 Conclusion and Future Work

In this paper we presented a serious problem that affects data sharing. Regarding this problem it was introduced a series of tools that will be used to solve it, namely Blockchain and Smart Contracts. Together with the usage of blockchain implementations like Hyperledger Fabric or Ethereum Project, are to be taken advantage in the proposed work. Regarding related work, namely Sia and Storj, they provide insight on how these services handle similar issues. Therefore, what is stated in this paper will present basis of the future work to be done.

At the current state, the topic of blockchain has gone through a process of analysis in order to understand its capabilities and maximize its use. Additionally, existing blockchain related work and existing blockchain implementations (Hyperledger Fabric and Ethereum Project) have also been analyzed to better understand how the technology can be used to solve this specific problem.

To do so, several approaches to the problem were analyzed and described to better understand how it can be tackled. Lastly, the design of the network, the development of user applications, the development of smart contract rules will be improved in the future in order to reach a better solution to solve the problem presented in this paper.

In short, using the Blockchain as a form of safety mechanism, it is possible to to share data in a secure way and keep control of the rules and politics that are associated with data access.

The work presented in this paper will be the basis of the following iterations. Future work includes, but not limited to, better understanding of smart contracts and their full capabilities and restrictions; improvement of current users applications; the creation of a new smart contract as stated in Section 3.1 for file permissions and the creation of methods to impose restrictions on file sharing; the improvement the existing smart contract for file management to allow for a better file management; considering a distributed approach for file storage as stated in Section 2.4.

References

1. Ethereum project. `https://www.ethereum.org/`.
2. Godoc - shim package. `https://godoc.org/github.com/hyperledger/fabric/core/chaincode/shim`.
3. Sia. `https://sia.tech/`.
4. Storj. `https://storj.io/`.
5. Vitalik Buterin et al. Ethereum white paper, 2013.
6. Christian Cachin. Architecture of the hyperledger blockchain fabric. In *Workshop on Distributed Cryptocurrencies and Consensus Ledgers*, 2016.
7. Manav Gupta. *Blockchain for dummies*. John Wiley & Sons, 2017.
8. Hyperledger. hyperledger-fabricdocs documentation. Technical report, Linux Foundation, 2017.
9. Preethi Kasireddy. Ethereum how it works, anyway? `https://medium.com/@preethikasireddy/how-does-ethereum-work-anyway-22d1df506369`, 2017.
10. Antony Lewis. A gentle introduction to blockchain technology. *Brave New Coin*, 2015.
11. Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system.
12. David Vorick and Luke Champine. Sia: Simple decentralized storage. 2014.
13. Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Project Yellow Paper*, 151, 2014.

Markov random field-based prediction for mobile robots

Hongjun Li, Miguel Barão and Luís Rato

Departamento de Informática,
Universidade de Évora, Portugal
li.hongjun@foxmail.com,{mjsb,lmr}@uevora.pt

Abstract. When exploring unknown environments, mobile robots can do tasks better with more information. Mapping with known poses is a subtask in simultaneous localization and mapping (SLAM). This paper focuses on how to predict the map and proposes a new method for mobile robots with known poses. This method is based on classical occupancy grid mapping and Markov random field (MRF). A grid map is used to represent the environment. The value of every grid cell in this map is its occupancy probability in log odds form and extended into continuous space from a binary value. Occupancy grid mapping is applied to deal with the raw observed data. An MRF model is built to recover the surface of the map. The results of occupancy grid mapping in log odds form are the observation of the MRF model. In continuous space, the optimal solution of the MRF can be obtained by solving an inverse matrix. When new data is obtained, the inverse matrix can be computed recursively.

Keywords: Occupancy grid mapping, log odds form, Markov random field

1 Introduction

In occupancy grid mapping [1], a map is divided into many grid cells and these grid cells are assumed to be independent of each other. Binary Bayes filter [2] can be applied to do the mapping task recursively. In log odds form, binary Bayes filter is additive. Occupancy grid maps are convenient for robot navigation and other further tasks. It has been developed in robot mapping problem by many researchers.

Some prediction methods have been proposed. In [3], occupancy grid map is adopted to represent the environment and prediction-based SLAM algorithm (P-SLAM) is proposed. By collecting the structure information surrounding unknown space and matching the structures, the structures of unobserved space can be predicted. This idea is extended in [4] to predict map and the motion of moving objects simultaneously. A topological map is applied to predict map structure. Similarly, [5] predicts unexplored areas by finding similarities between the current surroundings of unobserved space and previously built maps. This method can help mobile robots to explore unobserved space. Gaussian process

occupancy map (GPOM) [6] is different from the above prediction methods and there is no need to compare sub-maps or structures. The map is regarded as a Gaussian random field and occupancy mapping is a binary classification problem based on the occupied or free points in observed space.

In this paper, the values representing the map are extended to $(-\infty, +\infty)$ from binary values and MRF [7] is applied to predict the map. In section 2, occupancy grid mapping is introduced. An MRF model of the map is built in Section 3. For every grid cell, the value is the log odds form of the occupancy probability. The observation of the MRF is the result of occupancy grid mapping in log odds form. In Section 4, maximizing the posterior distribution of the MRF model, a linear equation set is obtained. By solving the inverse matrix, the optimal solution can be obtained. Finally, a simulation is done in Section 5.

2 Occupancy grid mapping

In standard occupancy grid mapping, the grid cells are assumed to be independent of each other. For every grid cell m_i, it has two possible states: occupied and free. The corresponding probabilities are denoted by $p(m_i)$ and $p(\bar{m}_i)$, respectively. This is a binary estimation problem. The odds of the occupancy state is defined as

$$\frac{p(m_i)}{p(\bar{m}_i)}, \tag{1}$$

and the log odds form [2] is defined as

$$\log \frac{p(m_i)}{p(\bar{m}_i)}. \tag{2}$$

The range of the log odds form is $(-\infty, +\infty)$. The log odds form can be transformed into occupancy probability by the sigmoid function.

Assume the observation set is $z_{1:t} = \{z_1, z_2, ..., z_t\}$. Based on Bayes rule, every grid cell can be done individually as

$$p(m_i|z_{1:t}) = \frac{p(z_t|m_i)p(m_i|z_{1:t-1})}{p(z_t|z_{1:t-1})}, \tag{3}$$

where

$$p(z_t|z_{1:t-1}) = p(z_t|m_i)p(m_i|z_{1:t-1}) + p(z_t|\bar{m}_i)p(\bar{m}_i|z_{1:t-1}), \tag{4}$$

$p(m_i|z_{1:t})$ is the posterior probability distribution and $p(z_t|m_i)$ is the measurement probability conditional on the grid cell m_i at time t. By analogy, we have

$$p(\bar{m}_i \mid z_{1:t}) = \frac{p(z_t \mid \bar{m}_i)p(\bar{m}_i \mid z_{1:t-1})}{p(z_t \mid z_{1:t-1})}. \tag{5}$$

Combining equation(3) and (5), the odds form of $p(m_i|z_{1:t})$ is formulated as

$$\frac{p(m_i \mid z_{1:t})}{p(\bar{m}_i \mid z_{1:t})} = \frac{p(z_t \mid m_i)}{p(z_t \mid \bar{m}_i)} \frac{p(m_i \mid z_{1:t-1})}{p(\bar{m}_i \mid z_{1:t-1})}. \tag{6}$$

In the same manner, the odds forms of $p(m_i \mid z_{1:t-1})$, $\cdots$, $p(m_i \mid z_1)$ can be obtained. Replacing the same items in equation (6) recursively, equation (6) be rewritten as

$$\frac{p(m_i \mid z_{1:t})}{p(\bar{m}_i \mid z_{1:t})} = \frac{p(z_t \mid m_i)}{p(z_t \mid \bar{m}_i)} \cdots \frac{p(z_1 \mid m_i)}{p(z_1 \mid \bar{m}_i)} \frac{p(m_i)}{p(\bar{m}_i)}. \tag{7}$$

Its log odds form is formulated as

$$\log \frac{p(m_i \mid z_{1:t})}{p(\bar{m}_i \mid z_{1:t})} = \log \frac{p(z_t \mid m_i)}{p(z_t \mid \bar{m}_i)} + \cdots + \log \frac{p(z_1 \mid m_i)}{p(z_1 \mid \bar{m}_i)} + \log \frac{p(m_i)}{p(\bar{m}_i)}. \tag{8}$$

In log odds form, the Bayes filter is additive. If the log odds occupancy observation $\log \frac{p(z_t|m_i)}{p(z_t|\bar{m}_i)}$ is known, Bayes filter can computed efficiently

3 The MRF Model

The map is regard as a MRF represented by $l = [l_i]^{\mathrm{T}}$, where $l_i = \log \frac{p'(m_i)}{p'(\bar{m}_i)}$ is the 'true' value underlying the observation $O_i = \log \frac{p(m_i|z_{1:t})}{p(\bar{m}_i|z_{1:t})}$. A second-order neighbourhood system [7], which includes the diagonal grid cells, is shown as Figure 1. Every gird cell has 8 neighbours. The neighbours are denoted by $l_{i'}$.

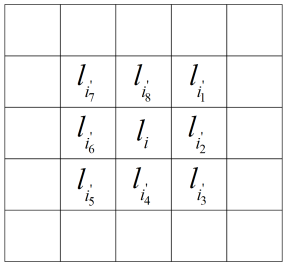


Fig. 1: MRF model

A clique c is defined as a subset of variables that are neighbours to one another. The pair-variable cliques are shown in Figure 2. The collection of the pair-variable cliques is denoted by C_2.

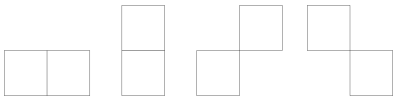


Fig. 2: Pair-variable cliques in second-order neighbourhood system

3.1 Prior Probability

Here only pair-variable cliques in the second-order neighbourhood system are considered. The prior probability is formulated as

$$p(l) = \frac{1}{Z}\exp(-\sum_{c\in C_2} V_c(l)), \tag{9}$$

where

$$Z = \sum_{l}\exp(-\sum_{c\in C_2} V_c(l)). \tag{10}$$

and $V_c(l)$ is the clique potential of clique c. The clique potential is defined as

$$V_c(l) = (l_i - l_{i'})^2, \tag{11}$$

which is based on the deviation of two log odds occupancy probabilities. Expanding all the clique potentials, the sum is quadratic. The prior distribution (9) can be rewritten as

$$p(l) = \frac{1}{Z}\exp(-2l^{\mathrm{T}}\mathcal{A}l), \tag{12}$$

where $\mathcal{A}$ is a circulant matrix. There are a log of maximums in the prior distribution. As a result, $\mathcal{A}$ is singular.

3.2 Likelihood

Assume the observation vector is denoted by $O = [O_i]^{\mathrm{T}}$. If l_i has no observation, the observation O_i is 0. The observations are independent of each other. The likelihood is based on the deviation between the log odds occupancy probability and the corresponding observation. Assume the index set of the observed grid cells are denoted by $\mathcal{I}$. The likelihood can be formulated as

$$p(O|l) = \frac{1}{\prod_{i\in\mathcal{I}}\sqrt{2\pi\sigma_i^2}}\exp(-\sum_{i\in\mathcal{I}}(l_i - O_i)^2/2\sigma_i^2). \tag{13}$$

The likelihood can also be rewritten as

$$p(O|l) = \frac{1}{\prod_{i\in\mathcal{I}}\sqrt{2\pi\sigma_i^2}}\exp(-(l-O)^{\mathrm{T}}\Lambda'(l-O)/2), \tag{14}$$

where Λ' is a diagonal matrix. If one grid cell is not observed, it is not considered in the likelihood and the corresponding element in Λ' is 0.

3.3 Posterior Probability

Based on Bayes rule, the posterior distribution is obtained as

$$P(l \mid O) = \eta p(O \mid l)P(l) = \eta\frac{1}{\prod_{i\in\mathcal{I}}\sqrt{2\pi\sigma_i^2}}\frac{1}{Z}\exp(-E(l)), \tag{15}$$

where η is a constant and

$$E(l) = 2l^{\mathrm{T}}\mathcal{A}l + (l-O)^{\mathrm{T}}\Lambda'(l-O)/2. \tag{16}$$

4 Prediction

The mapping result can be obtained by maximizing the posterior distribution $p(l|O)$ or minimizing the posterior energy function $E(l)$ equivalently. The derivative of $E(l)$ with respect to l is formulated as

$$\frac{\mathrm{d}}{\mathrm{d}l}E(l) = 4\mathcal{A}l + \Lambda(l - O). \tag{17}$$

Let the derivative $\frac{\mathrm{d}}{\mathrm{d}l}E(l)$ be zero, a linear equation set is obtained and formulated as

$$(4\mathcal{A} + \Lambda')l = \Lambda' O. \tag{18}$$

When there is no observation, $\mathcal{H}' = 4\mathcal{A}+\Lambda'$ is singular and there is no solution to this linear equation set. When there is one observation, $\mathcal{H}'$ becomes nonsingular and the solution is formulated as

$$l = \mathcal{H}'^{-1}\Lambda' O. \tag{19}$$

If one new grid cell is observed and the corresponding variance is σ^2, Λ' can be rewritten as

$$\Lambda' + \mathrm{diag}(\cdots, 0, 1/\sigma^2, 0, \cdots) = \Lambda' + \mathfrak{b}\mathfrak{d}^{\mathrm{T}}, \tag{20}$$

where $\mathfrak{b} = [\cdots, 0, 1, 0, \cdots]^{\mathrm{T}}$ and $\mathfrak{d} = [\cdots, 0, 1/\sigma^2, 0, \cdots]^{\mathrm{T}}$. $\mathcal{H}'$ becomes $\mathcal{H}' + \mathfrak{b}\mathfrak{d}^{\mathrm{T}}$. Based on Sherman-Morrison equation, the inverse matrix can be formulated as

$$(\mathcal{H}' + \mathfrak{b}\mathfrak{d}^{\mathrm{T}})^{-1} = \mathcal{H}'^{-1} - \frac{\mathcal{H}'^{-1}\mathfrak{b}\mathfrak{d}^{\mathrm{T}}\mathcal{H}'^{-1}}{1 + \mathfrak{d}^{\mathrm{T}}\mathcal{H}'^{-1}\mathfrak{b}}. \tag{21}$$

When more grid cells are observed, the inverse matrix can be computed recursively.

If the map size does not increase, the inverse operation can only be done once. When the map size is very big, it is not easy to get the inverse matrix of $\mathcal{H}'$ at the beginning. When all the variances of the observations are the same as σ^2, the final prediction result can be computed in another way. At the beginning, all the observations are assumed to be obtained and $\mathcal{H}'$ is rewritten as $4\mathcal{A} + \Lambda$. The prediction becomes a filter and the inverse matrix $\mathcal{H}'^{-1}$ can be constructed easily as our previous work [8]. If one grid cell is not observed, we need to remove the corresponding variance σ^2 from Λ. $\mathcal{H}'$ becomes $\mathcal{H}' - \mathfrak{b}\mathfrak{d}^{\mathrm{T}}$. Based on Sherman-Morrison equation, the inverse matrix can be formulated as

$$(\mathcal{H}' - \mathfrak{b}\mathfrak{d}^{\mathrm{T}})^{-1} = \mathcal{H}'^{-1} + \frac{\mathcal{H}'^{-1}\mathfrak{b}\mathfrak{d}^{\mathrm{T}}\mathcal{H}'^{-1}}{1 - \mathfrak{d}^{\mathrm{T}}\mathcal{H}'^{-1}\mathfrak{b}}. \tag{22}$$

The step should be done again and again until all the unobserved grid cells are removed.

5 Simulation

The true map and the trajectory are shown as Figure 3. There are some walls and objects. The robot runs from ① to ② and there are two measurement directions: $\pm\pi/4$. The maximum range is 15 grid cells. If the measurement range is not the maximum range, the grid cell at the end of the measurement is occupied with log odds occupancy 9 and the other grid cells in the measurement range are free with log odds occupancy -7. If the measurement range is the maximum range, all the grid cells in the measurement range are free. The initial log odds occupancy is 0 and the observations are shown in Figure 4. The data shown in Figure 4a is used as the observations of the MRF model. The corresponding probabilistic occupancy observation is shown as Figure 4b. Because of the noise of the sensors, the borders of the objects and the walls are not observed clearly.

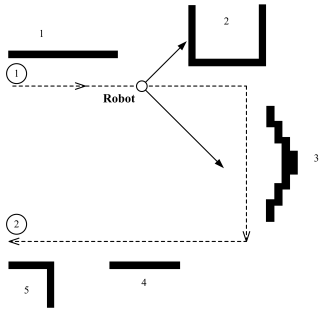
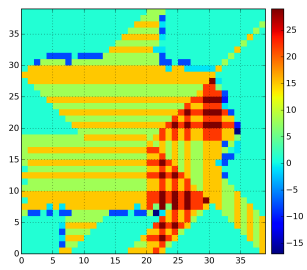
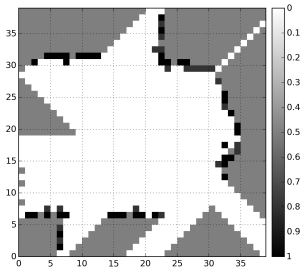


Fig. 3: The true map and trajectory



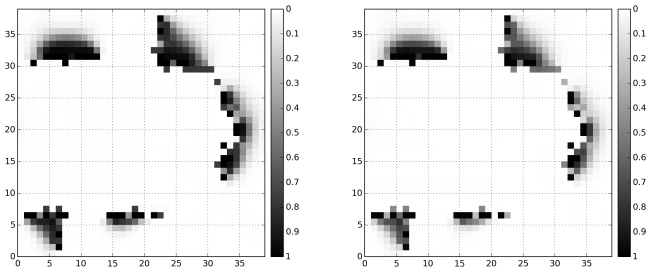
(a) The observations in log odds form



(b) The probabilistic occupancy observations

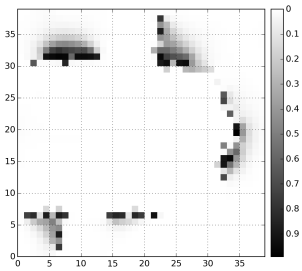
Fig. 4: The observations

The predictions with different σ^2 shown as Figure 5. Based on the observed borders of the objects, the prediction can construct the shape of the objects. The free space around wall 4 is more than that around wall 1 in the observation, wall 4 spreads less space in the prediction. In the MRF, neighbours usually have similar occupancy probabilities. Most of the observed space is free. As a result, most of the unobserved space are predicted free and only the unobserved space behind the occupied space are predicted occupied.

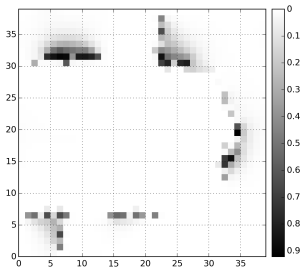


(a) $\sigma^2 = 0.001$

(b) $\sigma^2 = 0.01$



(c) $\sigma^2 = 0.03$



(d) $\sigma^2 = 0.05$

Fig. 5: Predictions with different σ^2

The size of this map is not very big, the prediction can be solved by computing equation (19) in several seconds. Meanwhile, the recursive method takes about one minute. When the map size is very big, the inverse matrix can not be computed directly, the recursive method can be applied. The disadvantage is that it takes a long time.

6 Conclusion

In this paper, we propose a new method to predict the. The map is regarded as an MRF and the values of the random variables are the occupancy probabilities in log odds form. The observations are computed by binary Bayes filter in

occupancy grid mapping. The best estimation can be obtained by solving a linear equation set. Based on the Sherman-Morrison equation, the problem can be solved recursively when new grid cells are observed. When the map size is very big, this method cannot be implemented online. In the future, we will improve this work to be one online method and implement it in exploring task.

Acknowledgment

This work was supported by EACEA under the Erasmus Mundus Action 2, Strand 1 project LEADER - Links in Europe and Asia for engineering, eDucation, Enterprise and Research exchanges. This work has been done in the scope of "PhD Seminar III course".

References

1. Alberto Elfes. Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57, 1989.
2. Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT press, 2005.
3. H Jacky Chang, CS George Lee, Yung-Hsiang Lu, and Y Charlie Hu. P-slam: Simultaneous localization and mapping with environmental-structure prediction. *IEEE Transactions on Robotics*, 23(2):281–293, 2007.
4. Shu Yun Chung and Han Pang Huang. Simultaneous topological map prediction and moving object trajectory prediction in unknown environments. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1594–1599. IEEE, 2008.
5. Daniel Perea Ström, Fabrizio Nenci, and Cyrill Stachniss. Predictive exploration considering previously mapped environments. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2761–2766. IEEE, 2015.
6. Simon T OCallaghan and Fabio T Ramos. Gaussian process occupancy maps. *The International Journal of Robotics Research*, 31(1):42–62, 2012.
7. Stan Z Li. *Markov random field modeling in image analysis*. Springer Science & Business Media, 2009.
8. Hongjun Li, Miguel Barao, and Luis Rato. Online learning occupancy grid maps for mobile robots. In *Workshop on Sustainability and Green Technology*, 2017.

Sentiment Analysis in Twitter: Experiments using Convolutional Neural Networks

Enkhzol Dovdon, José Saias, and Teresa Gonçalves

DI - ECT - Universidade de Évora
Rua Romão Ramalho, 59
7000-671 Évora, Portugal
d36506@alunos.uevora.pt,jsaias@uevora.pt,tcg@uevora.pt

Abstract. The paper describes a target-oriented message polarity classification system for three classes (positive, negative, and neutral) in Twitter. We evaluated some experiments using Convolutional Neural Networks (CNN) with different hyperparameters, TensorFlow library, Natural Language Processing toolkits, and a sentiment lexicon for tweet datasets. The best experiment has achieved 0.656 for the average recall rate, 0.678 for the average precision rate, 0.66 for the average F1 score, and 0.795 for accuracy.

Keywords: target-oriented sentiment analysis, opinion mining, sentiment analysis, text classification

1 Introduction

Demand for text analysis is dramatically rising from social networks. Twitter is one of the popular social networks. Sentiment analysis is the field of study that analyzes opinions, sentiments, and emotions of people towards entities and their attributes expressed in a written text [1]. Target-oriented (target-based, entity based, or topic based) sentiment analysis focuses on a sentiment polarity of given target.

This paper describes a system for target-based message polarity classification (positive, negative, or neutral sentiment) of a given message (tweet) towards that target (topic). For example: Given message: *"Sif's crush on Thor is adorable but also sad she should just hook up with may"*, Target: *"Thor"*, Classified polarity: *"positive"*. A system of Convolutional Neural Networks with one-layer of convolution on word vectors was developed using TensorFlow library [2] and Natural Language Processing (NLP) toolkits. In this work, our aim is to identify empirically the best settings for one-layer CNN. We focused on some of the settings such as the effect of combining different filter region sizes, various activation functions, word embedding techniques, and experimented with data pre-processing.

The rest of the paper is structured as follows: In Section 1, an introduction to the work is explained. In Section 2, some state-of-the-art of CNN are written. In Section 3, a classifier model and detailed information of the model are

presented. In Section 4, the entire datasets, hyperparameters of CNN, and data pre-processing are shown. In Section 5, the experimental results are presented. Finally, the conclusions, as well as further work are described in Section 6.

2 Related work

In this section, we briefly review the research works of sentiment analysis in Twitter based on CNN. Researchers proposed various features to classify text using CNN with different methods, including unsupervised and supervised classification.

Zang and Wallace [3] conducted an extensive experimental analysis of one-layer CNN, Support Vector Machine (SVM), and logistic regression for nine sentence classification datasets. They aimed to determinate empirically the valuable settings for CNN and provide a reasonable range for each hyperparameter: input word vector representations (word2vec and GloVe), different filter region sizes, the number of feature maps (a range of 10 to 2000), seven activation functions (ReLU, hyperbolic tangent, Sigmoid, SoftPlus, Cube, tanh cube, and Iden), the pooling strategy (k-max pooling strategy), and regularization terms (varying the dropout rate from 0.0 to 0.9 and fixing the l2 norm constraint to 3). They used a tokenized sentence that it was converted to a sentence matrix as an 'image' for convolution inputs and optimization is performed using SGD and back-propagation [4]. In their experiments, the best activation functions are ReLU, tanh, and Iden; 1-max pooling strategy was performed well; and the filter region size and the number of feature maps can have a large effect on performance.

Zhang, Zhao, and LeCun [5] worked character-level convolutional networks in several large-scale datasets and evaluated traditional models such as bag of words, n-grams and their TFIDF variants, word-based CNN and recurrent neural networks. They concluded that character-level CNN could work for text classification without the need for words from their experiments.

3 Model

A one-layer CNN architecture, shown in figure 1, was used for the system. CNN, are a specialized kind of neural network for processing data, use *"convolution"* mathematical operation in place of general matrix multiplication in one or more of its layers [6]. A model was implemented similarly to Kim Yoons Convolutional Neural Networks for Sentence Classification [7]. An example of Denny Britz [8] with TensorFlow library was also used for the system.

Convolution is a mathematical operation that is to process an input data such as embedded text or image using specified one or more filters with weights, then gives next layer feature maps (convoluted feature) as output. A convolutional layer has the major components: filters, parameter sharing, layer-specific hyperparameters, and activation maps [9]. The following major components were utilized for the experiments: filter sizes from 2 to 6, embedding character size of *128*, and strides of *[1, 1, 1, 1]*. Rectified linear unit (ReLU) as *F(x) = max(0,x)*

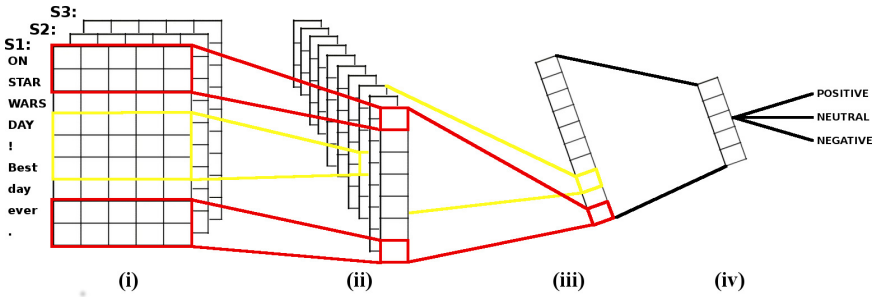


Fig. 1. Model architecture for an example sentence based on Kims study [7]. S: sentence (tweet). (i) Data pre-processing: NxK representation of tweet, (ii) Convolutional layer with multiple filter widths and feature maps, (iii) Max-pooling and flattening layer, (iv) Fully connected layer with dropout and softmax output.

or ReLU6 as *F(x) = min(max(features, 0), 6)* were used for an activation function with *0.1* bias. ReLU, non-saturated activation function, is commonly used for a deep learning system. According to a study by Xu [10], the advantages of using ReLU are it avoids *"exploding/vanishing gradient"* problem and training of neural network is faster.

The max-pooling function was applied to the following arguments that are mostly similar to create a convolutional layer. There was also the size of the window for each dimension of the input tensor. It is called a *ksize [1, sequence length filter size + 1, 1, 1]*. The output of a convolutional layer is a multi-dimensional Tensor. This tensor is converted into a one-dimensional tensor using the reshape operation which uses 384 filters. Dropout keep probability of the experiments was *0.5*.

4 Datasets and Experimental setup

4.1 Datasets

For this work, we are using a dataset of 20505 tweets collected for the task (Sentiment Analysis in Twitter) of the International Workshop on Semantic Evaluation (SemEval) [11]. All tweets are annotated for polarity as a positive, negative, or neutral by the organization. In order to evaluate the developed system, a split of 70:30 % was as training and test sets maintaining the ratio of each class. Detailed information of datasets is presented in Table 1. An example tweet from datasets is *"680885520373813248< tab >Thor< tab >positive(annotated label)< tab >Sif's crush on Thor is adorable but also sad she should just hook up with may"*.

4.2 Hyperparameters

These hyperparameters are used in the model: 2 to 6 words as different filter sizes with 128 feature maps; positive, negative, and neutral classes; dropout rate

Dataset	All	Positive	Negative	Neutral
Training set	14352	10464	2807	1081
Test set	6150	4484	1203	463
Total	20502	14948	4010	1544

Table 1. A description of the dataset.

of *0.5*; l2 regularization lambda of *0.0*; mini-batch sizes of 64; and epoch number of 200 for all datasets.

4.3 Data pre-processing

The vocabulary processor of *TensorFlow* library, *word2vec* word embedding methods, and a sentiment lexicon were used for the data representation.

Vocabulary processor of *TensorFlow*. Tokens of each tweet were represented by an embedded vector with the dimensionality of character embedding of 128 using vocabulary processor of *TensorFlow* after loading the entire dataset. Vocabulary size of all training dataset is 28486 words that are embedded vectors from *1* to *28486*. Input vector size of the network is the length of the longest tweet. For sentences with fewer tokens, the remaining elements are zeroed. The longest length of a tweet sentence in entire datasets consists of 51 tokens. In the below example of a tweet, the token length is 13: *"[adamlambert ca n't wait to see you in milan in june ! ! love youuu xx]"* → *"[1 2 3 4 5 6 7 8 9 8 10 11 12 13 0 0 ... 0]"*

word2vec. These vectors were trained on 100 billion words from Google News with the dimensionality of 300 and the continuous bag-of-words architecture[1].

A lexicon. A sentiment lexicon was used to indicate the term's polarity. The lexicon has 2006 and 4783 positive and negative words respectively [12] and [13] which was used for data pre-processing of experiments 5 and 6.

5 Experiments and results

The four kinds of experiments were evaluated using the previously mentioned model with the toolkits and the hyperparameters.

5.1 Experiments with different data pre-processing

The various datasets are created using Stanford POS tagger and the lists of positive and negative words. Target words in a tweet were changed with a specific word as *"TARGET"* and *"ITSTARGET"* in datasets of experiments 2,3, and 7. If *"TARGET"* word appears in a sentence of datasets of the second experiment,

[1] https://code.google.com/p/word2vec/

it will be the same word to identify the specified word as *"TARGET"* and a *"target"* word. Hence, we changed *"TARGET"* to *"ITSTARGET"* in experiment 3. Several experimental results in data pre-processing phase are shown in Table 2.

Experiment	**Modified tweet sentence**
1	Sif's crush on Thor is adorable but also sad she should just hook up with may
2	Sif's crush on TARGET is adorable but also sad she should .. with may
3	Sif's crush on ITSTARGET is adorable but also sad she should ... with may
4	NNP_Sif POS_'s NN_crush IN_on NNP_Thor VBZ_be ... IN_with MD_may
5	NNP_Sif POS_'s NN_negative IN_on NNP_Thor VBZ_be JJ_positive ... MD_may
6	NNP POS NN_negative IN NNP VBZ
7	PREV.NNP.Sif ... ITSTARGET AFTER.VBZ.be ... AFTER.MD.may

Table 2. Preprocessed text in datasets for experiments 1-7.

In the results, the first and second training have performed low measures and replacing a target with a specified word as *"TARGET"* or *"ITSTARGET"* in data pre-processing influence lack of efficient for prediction in Table 3. The result of the seventh experiment with features focused before and after target position are lower than others. A combination of POS tag and a word has achieved better performance than others and recall and precision are also similar to the fourth experiment. Therefore, if we use POS tag for the sentiment classification effectively, it will affect well.

Experiment	**Accuracy**	**Recall**	**Precision**	**F1 score**
1	0.86	0.62	0.86	0.64
2	0.74	0.54	0.57	0.55
3	0.54	0.54	0.56	0.55
4	0.78	0.66	0.68	0.66
5	0.80	0.62	0.70	0.65
6	0.74	0.47	0.60	0.49
7	0.75	0.53	0.59	0.55

Table 3. Results of experiments 1-7.

5.2 Experiments with various filter sizes

The results of experiments in the test dataset with different filter sizes on creating a convolutional layer is presented in Table 4.

The original paper author used 3,4, and 5 filter sizes on the model for movie reviews. Three and four filter sizes (embedded words) on the model for tweet

Experiment	Filter sizes	Accuracy	Recall	Precision	F1 score
1	3,4,5	0.775	0.630	0.664	0.642
8	2,3,4,5	0.795	0.606	0.690	0.638
9	3,4,5,6	0.792	0.636	0.676	0.651
10	3,4	0.786	0.645	0.674	0.657

Table 4. Results of experiments 1, 8-10.

datasets have achieved higher measure scores and model training loss is also lower than other experimental results.

5.3 Experiments with different activation functions and filter sizes

As previously mentioned, ReLU and ReLU6 activation functions are used for the model building. Scores of the ReLU6 function used in an experiment are greater than another activation function in Table 5.

Experiment	Filter sizes	Activation function	Accuracy	Recall	Precision	F1 score
1	3,4,5	ReLU	0.775	0.630	0.664	0.642
11	3,4,5	ReLU6	0.796	0.633	0.684	0.655
12	3,4	ReLU6	0.789	0.632	0.670	0.648

Table 5. Results of experiments 1, 11, and 12.

5.4 Experiments with multiple word embedding techniques

The experimental results using simple vocabulary processor of TensorFlow and word2vec are shown in Table 6.

Experiment	Hyperparameters	Accuracy	Recall	Precision	F1 score
1	vocabulary processor; ReLU; 3,4,5 filter sizes	0.775	0.630	0.664	0.642
13	Word2vec; ReLU; 3,4,5 filter sizes	0.788	0.644	0.686	0.658
14	Word2vec; ReLU6; 3,4 filter sizes	0.795	0.656	0.678	0.66

Table 6. Results of experiments 1, 13, and 14.

6 Conclusions

We conducted an experimental analysis of one-layer CNN for sentiment analysis of positive, negative, and neutral classes in differently pre-processed Twitter

datasets. The filter region size can have a large effect on performance. Three and four filter window sizes for tweet sentence work well such as reducing training time and increasing performance. Varying activation function has relatively little effect on the performance of this model. However, ReLU6 has given better scores than ReLU activation function. Currently, the result of the 14th experiment with word2vec, three and four filter sizes, and ReLU6 activation function for 3 classes classification has achieved the best measures. As a further work, we propose the following:

- To study more state-of-the-arts using CNN
- To add the more convolutional layer that it gives a higher result
- To use more NLP tools and lexicons that they enrich the current system
- To improve text pre-processing because of several pre-processing techniques
- To explore using CNN with different techniques
- To improve results and sustain the obtained results.

Acknowledgments. This work has been done in the scope of *"PhD Seminar III course"* and was supported by EACEA under the Erasmus Mundus Action 2, Strand 1 project gLINK - Sustainable Green Economies through Learning, Innovation, Networking and Knowledge Exchange. We would also like to thank the LabInterop project, for providing the infrastructure. LabInterop is funded by *Programa Operacional Regional do Alentejo* (INALENTEJO).

References

1. Liu, B. (2015). Sentiment analysis: Mining opinions, sentiments, and emotions. Cambridge University Press., pp.1.
2. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... & Ghemawat, S. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467.
3. Zhang, Y., & Wallace, B. (2015). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. arXiv preprint arXiv:1510.03820.
4. Rumelhart, D. E., Geoffrey E.H., & Ronald J. W. (1988) Learning representations by back-propagating errors. Cognitive modeling, 5:3.
5. Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text classification. In Advances in neural information processing systems (pp. 649-657).
6. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.
7. Kim, Y. (2014). Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.
8. Britz, D. (2015). Understanding Convolutional neural networks for NLP. URL: http://www. wildml. com/2015/11/understanding-convolutional-neuralnetworks-for-nlp/(visited on 11/07/2015).

9. Gibson, A., & Patterson, J. (2016). Deep learning: a practitioners approach. To Appear, March (pp.130-133).
10. Xu, B., Wang, N., Chen, T., & Li, M. (2015). Empirical evaluation of rectified activations in convolutional network. arXiv preprint arXiv:1505.00853.
11. Nakov, P., Ritter, A., Rosenthal, S., Sebastiani, F., & Stoyanov, V. (2016, June). SemEval-2016 Task 4: Sentiment Analysis in Twitter. In SemEval@ NAACL-HLT (pp. 1-18).
12. Hu, M., & Liu, B. (2004, August). Mining and summarizing customer reviews. In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 168-177). ACM.
13. Liu, B., Hu, M., & Cheng, J. (2005, May). Opinion observer: analyzing and comparing opinions on the web. In Proceedings of the 14th international conference on World Wide Web (pp. 342-351). ACM.

Identity Management in Healthcare Using Blockchain Technology

João Santos, Pedro Salgueiro, and Vítor Nogueira

Universidade de Évora, Portugal
m39519@alunos.uevora.pt
pds@di.uevora.pt
vbn@di.uevora.pt

Abstract. Using the Blockchain technology a system can be created to provide several benefits over traditional methods being used in today's Health digital landscape. Costs and risks associated with these systems can be reduced and information can become transparent and trustworthy to all participants. In this article the technological foundations that enable this change are explored and analyzed. The Hyperledger Fabric Network with true private transactions and advanced security mechanisms was used to serve as the basis for this system. An application was created that uses smart contracts to manipulate the ledger. This system will be presented and its impact in Healthcare discussed.

Keywords: Blockchain, Health, Identity, Big Data

1 Introduction

Health is becoming more digital thanks to the widespread availability of computing devices. More and more medical records are stored on a digital format. For storing patient clinical data and their identity in a medical context, the Electronic Health Record (EHR) was created.

While all this information should benefit both patient and health professionals alike, it is not being handled in an effective manner due to problems caused, in part, due to the fragmentation of the patients identity that naturally occurs in today's Health Information Systems.

Health is an important topic, for everyone. Healthcare should strive to provide the best service it can for everyone and everyone should have access to a quality service. EHR are being generated at an ever increasing rate but most of the data is not used in a way that puts the patient's privacy and trust at the forefront.

The purpose of the work presented in this paper is to create and implement a Blockchain based system for Identity Management in the Healthcare domain. The patient will be able to manage his data and control its access. Such a system would be suited to handle the patient's identity, for example, in hospitals

or clinics and would be able to solve many problems in how data is traditionally handled in the Information Systems (IS) available in a regular medical environment.

Blockchain is known as the technology behind the Bitcoin Cryptocurrency, altough nowadays it is being used for many more purposes that are explored in the following sections, and its main design goal is to provide security and immutability to an agreed upon list of records.

A blockchain runs on a network of computers and the list of records is replicated in some manner depending on the Blockchain implementation. The first Blockchain was conceptualized as the public ledger for the Bitcoin cryptocurrency in 2008 by Satoshi Nakamoto, a pen name of, a still unknown to this day, individual or organization of individuals. The network was implemented in 2009 and many are now finding it has a much broader potential across many fields, with some implementations even resembling a programming platform to execute code in an autonomous manner. [Nak08]

A single universal way to identify a person in a given environment is clearly something we should strive towards as seen in, for example, the *Cartão do Cidadão*, a portuguese identification document that replaces four other identification documents, streamlining portuguese civilian identification. This also allows many businesses to tailor their services to this document making it easier on both parts and eliminating unnecessary costs and risks.

Electronic Health Records (EHR) have seen some progress made regarding the standards that allow for interoperability between different organizations thanks to the Health Level 7 (HL7) standard. While this standard is growing in use and is represented internationally, Portugal has just started the work required to implement it. [Hea17]

In an effort to make the identity of a patient more secure and transparent a Blockchain can be used to create a system that puts at the forefront of its design the patients, breaking conventions in traditional patient data handling.

In this article different Blockchain implementations are explored and related work in this field is presented. More precisely, in Section 2, a brief introduction to Blockchain is made followed by an introduction to its most prominent implementations. Then a number of real-world use cases of this technology in the healthcare field are explored. In Section 3 technical details of the system will be presented. Finally, in Section 4, some conclusions are observed regarding the change enabled by these advances.

2 Background

While Blockchain is not a new concept at this point, it is an evolving technology that is being used to solve old problems with new approaches. This section

will explore the Blockchain technology origins and history, some of its different implementations and a brief history to the identity problem is presented.

2.1 Blockchain Technology

A Blockchain can be many things. It can refer to the Bitcoin Blockchain, alternative implementations or forks of the Bitcoin Blockchain called Altchains or even platforms that allow execution of code in an autonomous manner, exactly as it was programmed, with no human intervention. It is a continuously growing list of records, written in the ledger, a structure where records are written, that is being replicated across a network of devices in opposition to having a single central record history, making it a good example of a distributed database. [Woo17]

The main design goal of the Blockchain is security and to fulfill this purpose it uses techniques such as cryptography and digital signatures to not only verify the authenticity of records but also read or write access to the network.

Unlike a conventional central data storage, where only a single entity keeps a copy of the underlying database, the ledger of the Blockchain is replicated across any number of nodes. Not every participant has the same ability to interact with the ledger and in this respect a Blockchain can be permissionless or permissioned. In a permissionless Blockchain every node of the network can write in the Blockchain whereas in a permissioned Blockchain only a select group of entities have access to writing in the ledger, making the permissioned version, by default, secure if the entities themselves are secure and considered trustworthy.

How does a permissionless Blockchain maintain security if every participant has access to writing on it, including potentially malicious parties?

Take for example the Bitcoin Blockchain that uses a peer-to-peer network to avoid meddling from a financial institution or a third party in a financial transaction. Given that participating nodes in the network can belong to different and often competing parties, there is no implied trust between them, so the Blockchain needs a mechanism to ensure the integrity of the ledger and prevent malicious meddling from interested parties or to avoid a central authority.[Bar17]

To solve this problem, consensus mechanisms are used differently, depending on its implementation, but having, at its core, a solution to create immutable records and ensure security. In Bitcoin Blockchain's case, consensus is reached by the longest chain rule where the longest chain not only serves as proof of the sequence of events witnessed, but as proof that it came from the largest pool of computing power.[Baa16]

While the first Blockchain was conceptualized as the public ledger for the Bitcoin cryptocurrency in 2008 by Satoshi Nakamoto and implemented in 2009, many are now using it as a foundation across many application areas such as

identity management, traceability and asset management. Thanks to the roaring success of Bitcoin and the increasingly apparent use cases that the Blockchain can provide, the public awareness of it is rising and it is quickly becoming a technological foundation in our economic and social systems.

2.1.1 Ethereum

Bitcoin is getting media coverage almost everyday and public awareness in cryptocurrencies in general is rising. Some people are considering cryptocurrencies and the Blockchain, to be essentially the same technology and, while that may have been somewhat true not so long ago, Blockchain technology is starting to be used in a plethora of ways.

Ethereum is an open-source platform based on the Blockchain technology that enables developers to build and deploy Decentralized Applications (*DAPPs*). Ethereum is being developed by the Ethereum Foundation and was first discussed by Buterin in 2013. Ethereum intends to provide a Blockchain with a built-in programming language that is used to create *Smart contracts*. [Woo17]

These contracts are used to describe the logic of any system that developers can imagine and, when created, can then be deployed to the Blockchain where they execute as "autonomous agents". Thanks to these tools it is safe to say that long gone are the days where building Blockchain applications required a complex background in coding cryptography, mathematics as well as significant resources.[Woo17,Blo17a]

Ethereum Blockchain is a permissionless Blockchain, and thus, it must have a consensus mechanism to ensure the validation process of every record and, in turn, ensure security and immutability. While other implementations of the Blockchain have different consensus mechanics, in Ethereum's case, all participants have to reach consensus over the order of all transactions that have taken place. If a definitive order cannot be established then a double-spend might have occurred.

2.1.2 Fabric

Hyperledger Fabric (HLF) is part of the Hyperledger project started in December 2015 by the Linux Foundation, and is an open-source developer-focused community of communities focused on the development of enterprise-grade, open-source Blockchain-based solutions. Fabric is an implementation of a Distributed Ledger Platform (DLP) under the Hyperledger umbrella. [Cac16]

HLF's initial commit was contributed by IBM and written in Go language. It is a permissioned Blockchain and its main design goal was to surpass previous Blockchain implementation limitations, such as, lack of true private transactions and confidential contracts.

This is achieved thanks to assigning peers in the network three distinct roles and by offering the ability to create channels each with its own private ledger. A peer can have the role of endorser, committer or consenter or sometimes multiple roles. HLF is intended as a foundation for developing applications in a modular fashion, opting for a plug-and-play approach to various components. [Hyp17b]

HLF, as discussed, also allows the creation of smart contracts which can be written in Chaincode. As this Blockchain's key operational requirement is privacy, true private transactions and confidential contracts can exist and are a great asset for a business environment where sensitive information is necessary and disclosed often. Thanks to its modular approach consensus protocols are no longer hard-coded and trust models can be repurposed.

2.1.3 Burrow

Hyperledger Burrow (HLB) is also part of the Hyperledger project and its development started in 2014 by Monax and sponsored by Intel. It is a permissionable smart contract machine written in Go and offers a modular Blockchain client with a permissioned smart contract interpreter built, in part, to the specification of the Ethereum Virtual Machine (EVM) and the client has, essentially, three main components, the consensus engine, the permissioned EVM and the Remote Procedure Call (RPC) gateway. [KMBD17,Hyp17a]

HLB has its own Consensus Engine, the Byzantine fault-tolerant Tendermint protocol. The Tendermint protocol is an open-source effort that allows high performance in solving the consensus problem and also has a flexible interface for building arbitrary applications above the consensus, as well as, a suite of tools for deployments and their management. [Buc16]

2.2 Identity in Healthcare

Originally records of a patient were stored in a physical format. Thanks to the advent of the computers more and more records are stored on a digital format and the Electronic Health Record (EHR) was created. This benefits handling of information between the patient and the medical professionals and medical institutions. But first we must discuss what is defined as identity in this specific case.

Identity is a construct that depends on the context it is inserted. Identity can be defined as the characteristics determining who or what a person is. Particularly in this paper we can define identity as the characteristics that determine who the patient is in the given Healthcare ecosystem they belong to, such as, the name, the age, the cellphone number, the gender and the birth date of the

patient. Electronic Health Records encapsulate this information in a digital format, however they are usually formatted according to the Information System they were designed to work with. Standards are a tool that enable interoperability.

Standards for EHRs were created and many failed to bring the much needed consensus that was required for interoperability between different Information Systems in different institutions. Health Level 7 has done much work to be recognized in many countries and is quickly being implemented in many countries to allow for joint efforts between organizations.

Even with these advances in mind, the nature of many clinics and hospitals Information Systems makes the management of their patients identity a very cumbersome, costly and risky affair to handle. Security in a connected age, where internet is easily available, is lagging behind and presenting some problems. There is also the question of transparent use of information by the organizations that store it.

2.3 Blockchain for Identity Management in Healthcare: Use Cases

Some companies have already started developing Blockchain applications in the Healthcare field and established some key partnerships.

Many Blockchain-based solutions are still very early on development or deployment. One exception is Guardtime, that has fully deployed their system in 2008, started cooperating in 2011 and in 2016 announced a partnership with the Estonian Government, where a million patient records are now secured by the strategy and, until today, still proves the resilience of the Blockchain technology, as well as, other advances in cryptography. Now other companies like Verizon are becoming interested in this technology for their own purposes. [Gua18,Est16]

Another company, Gem, is collaborating with Phillips Healthcare to explore options in this area, and is opting to solve the interoperability problem with an additional layer of abstraction they call GemOS. Factom, another Blockchain-based service, has also announced a partnership with a major US medical services provider HealthNautica.[Blo17b,Fac17]

The use of the Blockchain technology in the health field is expanding. Just recently a new platform appeared, called Medichain that allows patients to store their own data in a secure way and give anonymized access to this data to specialists. Giving data allows for users to gain tokens that represent value. [Med18]

3 A HLF Network for Healthcare

Altough many use cases were presented in Section 2 they do not have the technology that allows true private contracts and some of them use a currency as a means to ensure the network consensus is achieved, by monetizing the patients data. After analyzing the different Blockchain implementations the Fabric network was chosen as the foundation for this system. This implementation was chosen to overcome previous works weaknesses and to match the security and privacy that patients expect to have with their health data. First we will discuss the tools provided to configure the network, and then discuss the system architecture.

3.1 Fabric Network Configuration Tools

Hyperledger Fabric (HLF) does not use a centralized ledger where every record is available to every participant in the network. Instead it opts to allow multiple ledgers in a network to achieve different goals of a greater purpose. This allows the creation of channels of information between trusted parties, for example, a channel of secure and private information between the clinical staff of an hospital and a patient.

A HLF network is comprised by the *cryptogen*, *configtxgen*, *configtxlator* and *peer* tools that are used to configure the network.

The *cryptogen* tool generates cryptographic data consuming the file *crypto-config.yaml*. HLF uses an abstraction layer for certification and authority called Membership Service Provider (MSP) that defines the rules by which entities are governed and authenticated and it must be unique for every participating entity.

The *configtxgen* tool generates the genesis block for the orderer services and the initial transactions. This tool consumes the file *configtx.yaml* that defines configuration parameters for channels, the genesis block and the orderer service.

The *configtxlator* tool is also used to generate channel configurations. Finally the *peer* tool is used to manage the participating peers in the HLF network.

These tools are used to create and maintain the topology of the network and are invoked when a change to the network is made, for example, when permissions to certain records are changed or a new user is enrolled in the network and are very much intertwined with the Certificate Authority (CA) server system to maintain the security that is needed in a sensitive subject that deals with private information.

3.2 Identity Representation in Hyperledger

HLF allows information to be written and read in a distributed manner with security and privacy at the forefront. Using smart contracts, a record is created to represent the concept of identity in this network.

The information that defines the patients identity is a key requirement to build a system that recognizes patients across the Healthcare environment, as discussed in Section 2. To this end, the identity of a patient is recorded on the ledger of the HLF network as a structure via a smart contract deployed to the network that interacts directly with the ledger. This structure contains the necessary fields to identify the patient such as its name and birth date, for example, as well as some other information necessary to manage this data.

To aid in interoperability with other systems, as seen in Figure 1, the Fast Healthcare Interoperability Resources (FHIR) standard by the Health Level 7 organization was used as basis for the representation of a patient. Each field of the structure that represents the patients identity, defined in the smart contract, is linked to a field of the patient structure as presented in the FHIR standard.

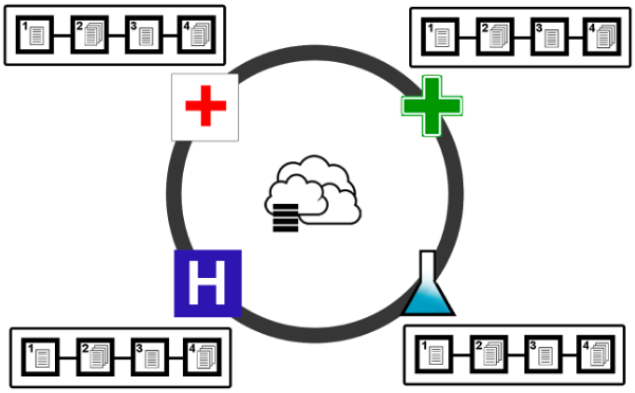


Fig. 1. An Example of Interoperability with the Blockchain Network

3.3 Application and Smart Contracts

To create an interactive system that can manage the patients identity in an Healthcare environment an application was built that the user interacts with. This application interfaces with smart contracts through the Hyperledger Fabric Software Development Kit and the chaincode was built using the Hyperledger Fabric Shim for node.js.

The application is accessed by the user and calls upon the smart contract. The smart contract will handle the assets part of the system. A smart contract to represent and manipulate identity was built and interfaces with the network to write and read records to the appropriate ledger. The overview of the architecture for this system is represented on Figure 2.

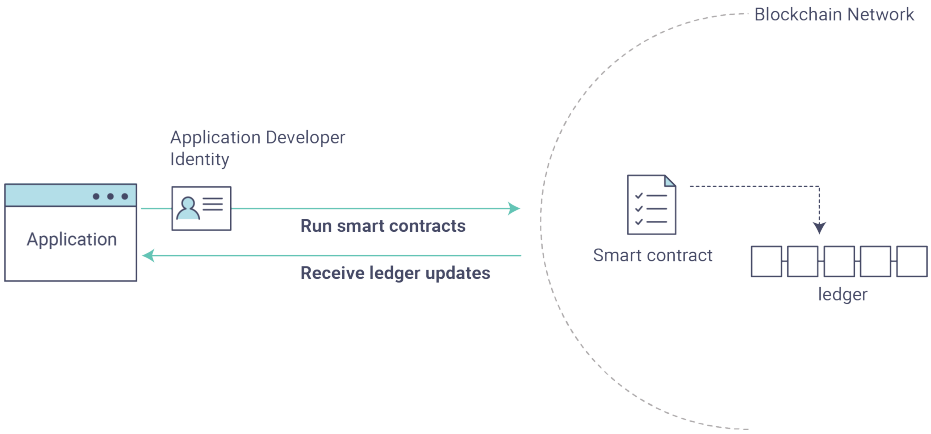


Fig. 2. An Overview of the System Architecture (Source: HLF Fabric Documentation)

The application allows for user enrollment to create a new identity in the network. When a new user of the application enters the network; the function, in the smart contract, that initializes the creation of the user and writes the user to the ledger as a new participating identity is called. Due to the security mechanisms this specific transaction is automatically signed by the administrator of the network and is verified by the CA servers.

The smart contract also provides the application with several operations to manage the identity object as seen on Figure 3. These operations form an Application Programming Interface (API) that return a payload in JSON format with identity information from the network. This API allows a query to be made to the network that returns the patients information, changing incorrect or outdated information or disabling the identity structure of someone who is not participating in the network actively anymore in order for that information to be read-only from that point on, for example, with more available. Depending on the operation only certain users can access the information or manipulate the already existing one. This system architecture leads to a modular as well as extensible approach regarding the availability of new operations that become available as soon as new versions of the smart contract are deployed.

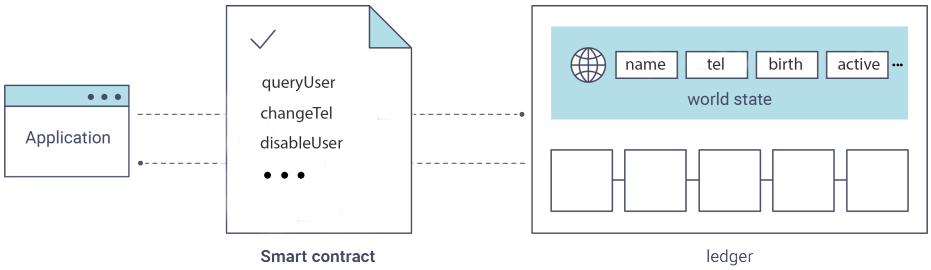


Fig. 3. Smart Contract Operations Example (Original: HLF Fabric Documentation)

4 Conclusion

In this document it was described that the way identity is handled by medical institutions nowadays presents a problem. Blockchain was explored as a tool to solve this problem and some of its different implementations were analyzed. Some practical use cases of this technology were also discussed. This research will enable solid foundations for future work.

It is safe to say that a system for improving the way a patient can interact with their health data can be built, using this technology, as discussed in previous sections. The system should allow for a transparent handling of personal data and be able to allow for secure management of access to this particular data.

If an advance is made in this regard it is expected that the patients trust in their Healthcare service is increased, and that risks and costs inherent to multiple independent information systems, that are not normalized to any standard, be reduced.

References

ACCG17. Daniel Augot, Hervé Chabanne, Olivier Clémot, and William George. Transforming face-to-face identity proofing into anonymous digital identity using the Bitcoin blockchain. pages 1–10, 2017.

Baa16. Djuri Baars. Towards Self-Sovereign Identity using Blockchain Technology. *University of Twente*, page 90, 2016.

Bar17. Iain Barclay. Innovative Applications of Blockchain Technology in Crime and Security. 2017.

Blo17a. BlockGeeks Ethereum Guide. `https://blockgeeks.com/guides/ethereum/`, 2017. [Online; Accessed November 29, 2017].

Blo17b. Is Blockchain the Answer to Healthcare's Big Data Problems? `https://healthitanalytics.com/news/is-blockchain-the-answer-to-healthcares-big-data-problems`, 2017. [Online; Accessed December 1, 2017].

Buc16. Ethan Buchman. Tendermint: Byzantine Fault Tolerance in the Age of Blockchains. 2016.

Cac16. Christian Cachin. Architecture of the hyperledger blockchain fabric. *IBM Research*, July, 2016.

Dre17. Daniel Drescher. *Blockchain Basics*. Apress, Berkeley, CA, 2017.

Est16. Estonian Government Adopts Blockchain To Secure 1 Mln Health Records. `https://cointelegraph.com/news/estonian-government-adopts-blockchain-to-secure-1-mln-health-records`, 2016. [Online; Accessed February 25, 2018].

Fac17. Factom's Latest Partnership Takes on US Healthcare. `https://cointelegraph.com/news/factoms-latest-partnership-takes-on-us-healthcare`, 2017. [Online; Accessed December 1, 2017].

Gua18. Verizon to Use KSI Blockchain Technology Developed for Estonia. `https://www.sdxcentral.com/articles/news/verizon-use-ksi-blockchain-technology-developed-estonia/2018/02/`, 2018. [Online; Accessed February 25, 2018].

Hea17. Health Level 7 Webpage. `https://hl7.org/`, 2017. [Online; Accessed January 15, 2018].

Hyp17a. Hyperledger Burrow Github Page. `https://github.com/hyperledger/burrow`, 2017. [Online; Accessed November 29, 2017].

Hyp17b. Hyperledger Fabric Documentation. `https://hyperledger-fabric.readthedocs.io/en/release/`, 2017. [Online; Accessed November 29, 2017].

KMBD17. Casey Kuhlman, Dan Middleton, Benjamin Bollen, and Silas Davis. Hyperledger Burrow (formerly eris-db). 2017.

Lew15. Antony Lewis. A gentle introduction to blockchain Technology. *Bits On Blocks*, pages 1–13, 2015.

Med18. MEDICHAIN - The Medical Big-Data Platform. `https://medichain.online/`, 2018. [Online; Accessed February 25, 2018].

Nak08. Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. *`https://bitcoin.org/bitcoin.pdf`*, page 9, 2008.

SWP16. David Shrier, Weige Wu, and Alex Pentland. Blockchain & Infrastructure (Identity , Data Security). pages 1–18, 2016.

VS17. Martin Valenta and Philipp Sandner. Comparison of Ethereum, Hyperledger Fabric and Corda. (June):1–8, 2017.

Vuk17. Marko Vukolić. Rethinking Permissioned Blockchains. *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts - BCC '17*, pages 3–7, 2017.

Woo17. Gavin Wood. Ethereum: a Secure Decentralised Generalised Transaction Ledger. 2017.

YL16. Affan Yasin and Lin Liu. An Online Identity and Smart Contract Management System. *Proceedings - International Computer Software and Applications Conference*, 2:192–198, 2016.

Statistical Recommender Systems Survey

Rodwan Bakkar Deyab, Irene Rodrigues

Universidade de Évora, Department of Informatics,
Rua Romão Ramalho n59, 7000-671 Évora, Portugal
d38745@alunos.uevora.pt, ipr@uevora.pt

Abstract. Recommender systems are used in a wide range of web applications like news, education, social media, tv streaming, e-commerce and job recruiting. In these web applications which contains huge number of items (news articles, movies, books, job offers, ...), it is not possible for the user to check all the items and he will not be interested in all of them also. These web applications use Recommender Systems to improve the user experience in finding the interesting items. Recommender Systems in the case of E-Learning is slightly different such that it recommends scientific content for the user and it recommends users to take specific scientific content considering there performance and knowledge. In this case the recommendation is reciprocal. Recommender Systems follow many approaches to achieve the recommendation task. It can be based on statistical methods of machine learning. It can also be based on ontologies which hold semantics about the domain of knowledge. In this work we survey works about recommender systems which were done using statistical methods. As recommender systems are used in the context of numerous items and resources, or in other words, in the context of big data, we survey some platforms and tools for big data processing.

Keywords: Recommender Systems, Machine Learning, Big Data, Parallel Processing, Big Data Processing Platforms

1 Introduction

Recommender systems nowadays are being more and more used in web applications that maintain huge data like in e-commerce[7], e-libraries[17] and e-tourism[14]. They improve the users experience in searching the items they need in the middle of millions of items. Recommender systems can work reciprocally like the case of e-learning[8] and job recruitment[21]. In these cases they can recommend useful content to the user as well as recommending users to the scientific content, the case of e-learning, or to the recruiters, the case of recruitment. Recommender systems make use of statistical methods to achieve their task. It can be done using knowledge-based methods also which needs the data to be represented in a semantical structure like ontologies. Dealing with big amounts of data in recommender systems, parallel processing comes as a perfect fit to elevate the performance and lessen the processing time. It can also be needed in

real-time scenarios. Big data is not only about the huge volume of the data as it may be understood by the name. Big data needs to have at least three characteristics to be specified as big data. They are called the three Vs in the literature. 3Vs are volume, velocity and variety [3]. The most obvious characteristic is the volume. Data is growing exponentially in volume. Velocity is not less important than volume to characterize the big data. The data of yesterday or even of the past minute can be out of date. It can be also a real-time data like the GPS system. Variety is the third most important characteristic of big data. Data can be in different formats like text, images and videos. This paper is organized as following: in section 2 we survey some distributed big data processing platforms. In section 3 we speak about the most important approaches in recommender systems. In section 4 we survey some recommender systems which use statistical methods and finally in section 5 we conclude the work.

2 Parallel Processing, The Great Fit for Recommender Systems

In typical cases, recommender systems deal with big data and the need for recommendation can be in the real-time. The recommendation process can be divided into smaller tasks. These smaller tasks can be ran to get partial results then these results can be merged to get the final result. For example, we have a dataset of millions of items and we need to compute the similarity between some item to the others. In this case we can divide and compute knowing that the input of any of the parallel tasks does not depend on the output of any other task. That is why parallel processing comes as a great fit for recommender systems[5]. Parallel processing allows the use of multiple machines together so we can benefit of all the resources these machines provide to improve the processing power and lessen the processing time. Many frameworks for parallel processing are used to process big data in recommender systems. Apache Spark is one of the state-of-art frameworks for this sake. It works in real time . Apache Spark is based on Apache Hadoop which provides fast batch processing of big data. Apache Storm is another real-time big data framework. We survey these three big data analytics platforms:

2.1 Big Data Analytics Platforms

Apache Hadoop It is a system of distributed processing of big data[16][1]. It is scalable so it can exploit thousands of machines in the cluster. It can only provide batch processing for the data. It has four modules:

Hadoop Common: Utilities to support other modules.

[1] http://hadoop.apache.org/

Hadoop Distributed File System (HDFS)[1]: It is a distributed file system built specifically for hadoop. It consistes of main node (master nodes) and one or more nodes (slave nodes) in the cluster. It divides the huge data among the nodes in the cluster so it does not need one big storage. Nodes can communicate with each other to replicate the data in case of failure of some node. The HDFS system has a second master which regularly connects to the main master and takes a copy of its directory and store it. The second master node serves in cases of the failure of the main master node. Figure 1 shows the architecture of the distributed file system of hadoop.

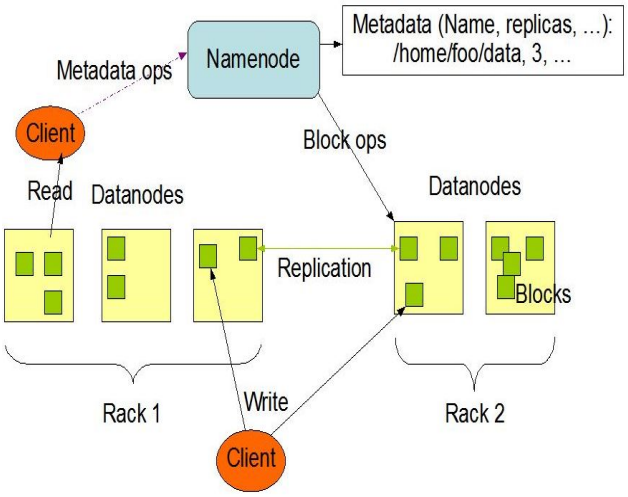


Fig. 1: The HDFS Architecture[2]

Hadoop MapReduce: A system for parallel processing of large data sets. It divides the huge data into chunks which are processed (mapped) to different nodes in the cluster. Then it receives the output of each processing node which will be an input of the reduce task. The processing will take place in the storage node such that hadoop pushes the processing to the node which contains the relevant data which makes it faster. A typical example about the MapReduce is the word count program shown in Figure 2. First it will split the data into chunks, second it will map the chunks to different working nodes to count then words then reduce the results to get the final counts.

[2] Source: https://hadoop.apache.org/docs/r2.6.0/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html
[3] Source: https://blog.jteam.nl/2009/08/04/introduction-to-hadoop/

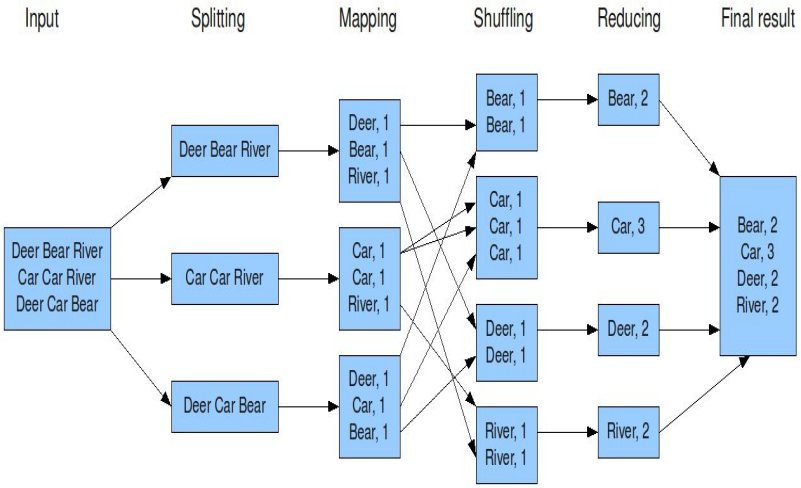


Fig. 2: The Word Count Example with the MapReduce Software[3]

Hadoop YARN[20]: is the new version of Hadoop MapReduce. Figure 3 shows the architecture of YARN. The ResourceManager has two main components: Scheduler and ApplicationsManager. The Scheduler allocates tasks to the nodes taking into consideration their computational capacities. The ApplicationsManager is responsible for accepting the job and assigning it to a specific container in the node machine.

Apache Storm It is a free and open source distributed real-time computation system[6]. Storm makes it easy to reliably process unbounded streams of data, doing for real-time processing what Hadoop did for batch processing[5]. The topology of storm is analogous to the MapReduce of apache hadoop. It consists of spouts and bolts exchanging streams of data. Storm's three important concepts are: streams, spouts and bolts:

Streams: Apache storm streams are tuples which can hold any type of data (integers, longs, shorts, bytes, strings, doubles, floats, booleans, and byte arrays). A stream in computing is a continuous sequence of bytes of data produced by one program and consumed by another program. It is consumed in a first in first out (FIFO) order. It can be bounded (finite) or unbounded (infinite). An example of a stream is the linux pipe.

```
cat logfile | wc -l
```

[4] Source: https://hadoop.apache.org/docs/r2.6.0/hadoop-yarn/hadoop-yarn-site/YARN.html

[5] http://storm.apache.org/index.html

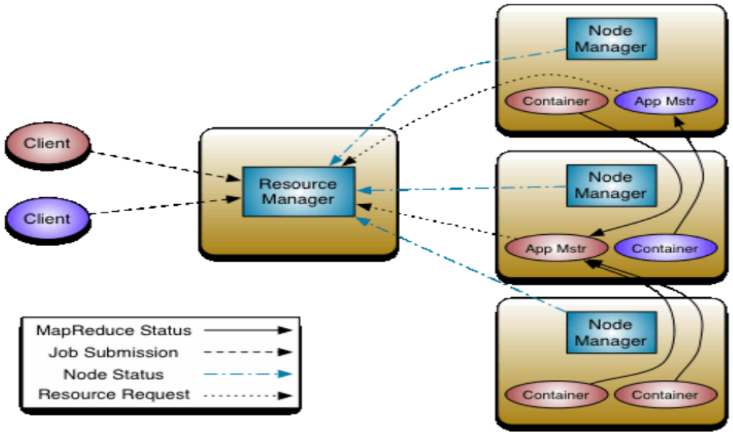


Fig. 3: The YARN Architecture[4]

The (cat) program reads the log file to be piped (streamed) to the word count (wc) program.

Spouts: Storm spouts process the external data to create streams as tuples and send them to bolts. It can get data from other queueing systems such as Kafka, Twitter etc. It can produce multiple streams of output.

Bolts: Storm bolts process the tuples from input streams and produce some output tuples. The input to bolts may come from spouts or from other bolts. Bolts can process any number of input streams. Input data is deserialized and the output data is serialized. Bolts run in parallel. Figure 4 shows the topology of Storm which is a group of spouts and bolts running parallelly in a Storm cluster

The structure of Apache Storm is a master-slave architecture. Figure 5 shows it in detail. The master server (nimbus) runs on a single node. Slave services called supervisors run on each worker node. Supervisors start one or more worker processes called workers that run in parallel to process the input. Worker processes store output to a file system or database. Zookeeper is used for the coordination of distributed processing.

Apache Spark It is a fast and general-purpose cluster computing system[22]. It provides high-level APIs in Java, Scala, Python and R, and an optimized engine that supports general execution graphs. It also supports a rich set of higher-level tools including Spark SQL for SQL and structured data processing, MLlib for

[6] Source: http://storm.apache.org/releases/current/Tutorial.html
[7] Source: https://jansipke.nl/installing-a-storm-cluster-on-centos-hosts/

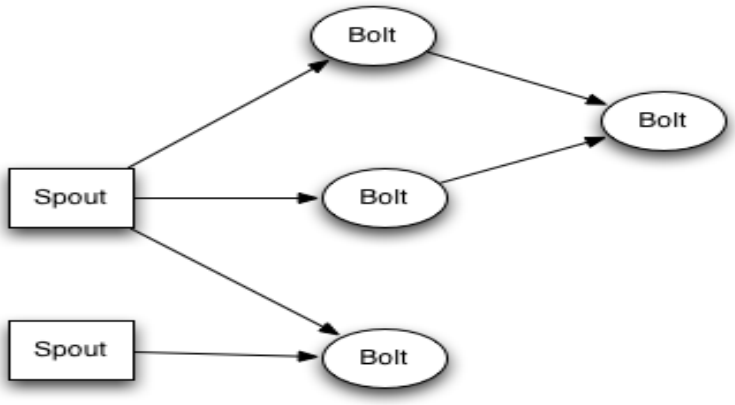


Fig. 4: Apache Storm Topology[6]

machine learning, GraphX for graph processing, and Spark Streaming[8]. Spark can achieve both purposes of Hadoop and Storm and it can analyze batch data and streams. Figure 6 shows the components of Apache Spark.

A Comparison of The Three Platforms Table 1 shows a comparison of the previous big data analytics platforms. The comparison takes into account some main aspects in big data processing.

	Apache Hadoop	**Apache Storm**	**Apache Spark**
ProcessingMethod	Batch processing	Real-time processing	Batch and Real-time processing
Fault-tolerance	Fault-tolerant	Fault-tolerant	Fault-tolerant
Performance	Fast and scalable	Slower than hadoop	Faster than hadoop
Distributed software	MapReduce	Sprouts and Bolts	Resilient Distributed Datasets (RDDs)

Table 1: A Comparison of The Three Platforms

3 Recommender System Approaches

Recommender system basically depends on the content of the data to recommend from, or on the users who are interacting with this data. In general, there are two main approaches for recommender system: content-based and collaborative filtering.

[8] https://spark.apache.org/docs/latest/
[9] Source: https://intellipaat.com/tutorial/spark-tutorial/apache-spark-components/

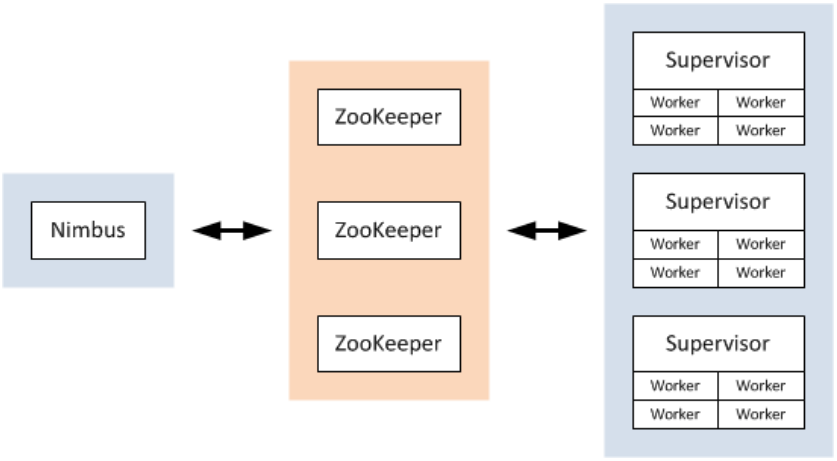


Fig. 5: Apache Storm Architecture[7]

Content-Based Filtering In this approach[15], the system will recommend items to the user similar to those items rated positively by him. The system will calculate similarity between items depending on some features of the items. This way lacks the ability of recommending various types of items to the user. For example, say that the user rates action movies highly, then the system will always recommend action movies to the user, it will not recommend drama or science fiction movies. This way of recommending is user-independent which means that the recommender system needs only to know about the user to whom it will recommend and does not need any information about other users as in Collaborative-Filtering. This way is able to recommend new items not rated by any user which is a problem in the Collaborative-Filtering approach. For new users which have not rated any items, this algorithm can not recommend any items. The recommender system represents the items by a set of features to be used in the recommendation process. For example, a movie item could be represented by the features: actors, genres, directors, etc. Figure 7 shows the content-based recommending approach:

Collaborative Filtering In this approach[2], the recommendations for a user are based on the preferences of other similar users. Just like in normal life people recommend to each other. There are two approaches:

User-Based Collaborative Filtering: In this approach[23], recommendations are calculated using the users which are similar to the user to whom we want to recommend. The items rated highly by similar users could be rated highly by the user to whom we want to recommend so we can recommend these items. The whole dataset is represented as a matrix and the similarity between users is calculated statistically. There are many algorithms to calculate the similarity

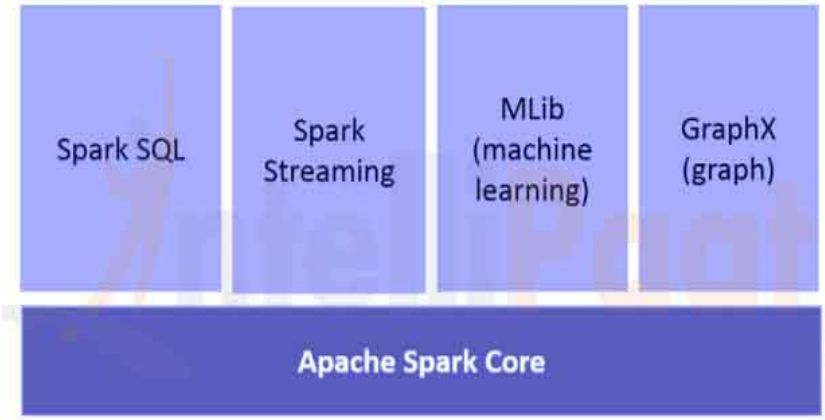


Fig. 6: Apache Spark Components[9]

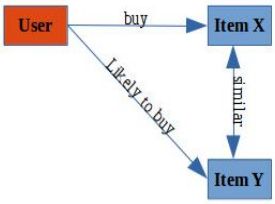


Fig. 7: Content-Based Filtering

between the users. Table 2 shows a dataset of users and items and the ratings of the users to the items.

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	8	1	?	2	7
User 2	2	?	5	7	5
User 3	5	4	7	4	7
User 4	7	1	7	3	8
User 5	1	7	4	6	5
User 6	8	3	8	3	7

Table 2: users-items-ratings dataset

So to recommend the item 3 or not to the user 1 we should look at the most similar users to the user 1 then recommend it or not according to the rating of the similar users to this item. Figure 8 shows the collaborative filtering approach.

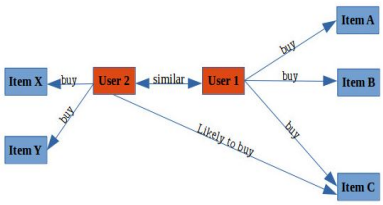


Fig. 8: Collaborative Filtering Approach

K Nearest Neighbors (KNN)[10] algorithm is the most used for collaborative filtering. ALS algorithm is another used algorithm, it will be explained in section 4. In the case of big datasets, it is not possible to handle the whole dataset so techniques of data reduction are used before the actual recommendations run. User-based Collaborative Filtering suffers from the cold-start problem. The cold-start is when the user is new he did not yet rate any items so we can not know the similar users to him. And the item cold start problem is when the item is not rated by similar users so we can not predict a rating for it, this problem does not appear in the content-based filtering. Another problem is the sparsity of the data in the big dataset with few rated items. Scalability is also a problem in the big datasets where computations become very expensive. User-based Collaborative Filtering is the most used approach for Collaborative Filtering and it shows better results and less complexity.

Item-Based Collaborative Systems[19, 13]: This approach is similar to the content-based approach in the sense that we calculate the similarity between items to achieve the recommendations. So if the user rates high some item then we can recommend similar items to this user. But it is different from the content-based approach in the sense that similarity between items is calculated in different way. In the content-based approach, we use features to calculate the similarity but in this approach, we compute it according to the ratings of these items by other users. This way is better scalable than User-Based Collaborative Filtering. It does not suffer from the cold-start problem but it suffers from the item cold-start problem such that there are not ratings for the new item to be able to compute the similarity between it and the other items.

4 Machine Learning Recommender Systems

Machine learning is a branch of Artificial Intelligence. It is to make the machine learn from data and discover models about it. Data is the fuel for the machine to learn. And when the machine receives new data, it can predict characteristics about it depending on the previous experience. Machine learning was defined by

Arthur Samuel back in 1959: Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed. Machine learning can be used in the context of big data and recommender systems. But in general we explain two approaches of machine learning:

Machine Learning Approaches

Supervised Machine Learning[11]: This approach is the most used. It depends on an annotated examples (a training set) to train and produce the predictive model. In the training set each example is annotated with the desired output value. Such that when the predictive model is built, it will assign as a prediction a value (a class) to the unseen example. Typical example about supervised machine learning is email spam detection which marks the received eamil as spam or not. Supervised machine learning has two types: classification and regression. Classification is to classify new data and assign predefined classes to it. Figure **??** explains the process of Classification.

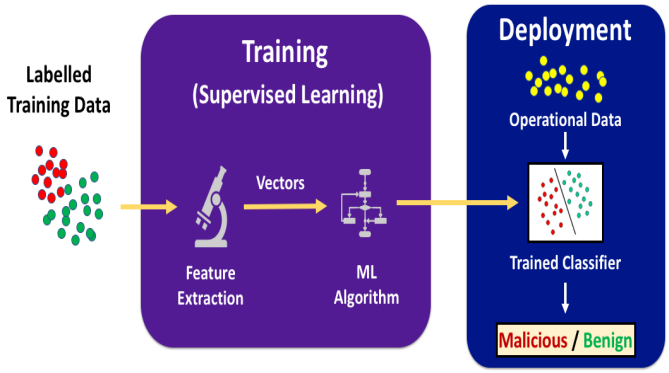


Fig. 9: Classification[10]

Regression is a statistical way to establish a relationship between a dependent variable and a set of independent variables. In other words, Linear Regression is a method to predict dependent variable (Y) based on values of independent variables (X). Examples are to predict house prices based on their properties and to predict how many units consumers will purchase. The difference between regression and classification is that the response of regression is continuous (some number) and the response of classification is categorical (specific classes). For example, when building a model to predict a continuous number between 0 and 10 then it is regression but if it was to predict yes or no then it is classification.

[10] Source: https://evademl.org/main/

Unsupervised Machine Learning[9]: The input data of this approach is unlabeled. It works on finding patterns to describe the distribution of this data. There is no method to evaluate the results as the data is not labeled. The k-means algorithm is a typical example about clustering as a typical problem in the unsupervised machine learning. One of the most used algorithms in recommender systems is:

Alternating Least Squares (ALS)[12]: ALS is a machine learning algorithm which can be used in Recommender Systems based on collaborative filtering approach. We explain this algorithm in a movie recommender context. ALS works on predicting the ratings of users on items which are not yet rated by these users. As this algorithm predicts the rating, we can recommend the items with the higher predicted rating value to the corresponding users. All data should be converted to numbers so we can run the algorithm. Then this algorithm expects the input as a matrix of users, items and ratings which is called the Utility Matrix. Table 3 shows an example of this matrix:

	Movie 1	Movie 2	Movie...	Movie N
User 1	1	BLANK	BLANK	3
User 2	BLANK	5	BLANK	3
User 3	BLANK	BLANK	1	BLANK
User 4	2	3	BLANK	BLANK
User 5	BLANK	BLANK	1	BLANK
User 6	4	BLANK	5	BLANK
User 7	BLANK	4	BLANK	BLANK
User...	BLANK	3	BLANK	BLANK
User M	BLANK	BLANK	BLANK	4

Table 3: Utility Matrix

Most users did not rate most items (movies) so ALS will predict these missing rating. Table 4 shows the predicted ratings and the ratings which can be used as recommendations.

	Movie 1	Movie 2	Movie...	Movie N
User 1	1	4	2	3
User 2	1	5	3	3
User 3	2.5	2.8	1	3.5
User 4	2	3	2	3.5
User 5	2.5	2.8	1	3.1
User 6	4	1.2	5	1.4
User 7	1	4	2.5	3
User...	2	3	2	3
User M	1	4	2	4

Table 4: ALS Predictions and Recommendations

We survey some of the recommender systems in some domains.

Recommender Systems Examples

- MyMediaLite[7]: *is a fast and scalable, multi-purpose library of recommender system algorithms.* It follows the collaborative filtering approach. It implements many algorithms like: K-Nearest Neighbor (KNN), and Alternating Least Squares (ALS) algorithms. Many experiments were held over the Movielens[11] and Netflix[12] datasets and showed good results.
- layer6.ai[21]: This system is the winner of the recsys challenge 2017[13]. It is a reciprocal recommender as it recommends job offers to users as well as users to recruiters. It was tested on the xing dataset which is a huge dataset with users, items and interactions between them. This system uses the xgboost library [4] which provides tree boosting machine learning mechanism.
- [8]: *works on the idea of recommending learning materials based on the similarity of content items (using Vector Space Model) and good learners' average rating strategy.*
- REJA[14]: is a system to recommend restaurants in a specific geographical area. It uses two methods of recommendations: collaborative filtering based on machine learning and knowledge-based method. It uses the K-Nearest Neighbors algorithm[10] in the first method.

Table 5 shows a comparison of the previous examples of recommender systems.

	Domain	Algorithms	Approaches
MyMediaLite[7]	E-Commerce, Movie,...	ALS, KNN,...	Collaborative Filtering
layer6.ai[21]	Job Recruitment	Tree Boosting	Content-Based Filtering
[8]	E-Learning	Vector Space Model[18]	Content-Based Filtering
REJA[14]	E-Tourism	KNN	Collaborative and Knowledge-Based

Table 5: Utility Matrix

5 Conclusion

In the world of big data, machines should be more aware of mining the valuable pieces of data and introduce them to the end user. Recommender Systems come into place to make this possible. Machine Learning is used in most cases to achieve the recommendation task. The need of recommendation in real-time is feasible thanks to the power of parallel processing and the nature of the recommendation process which can be divided on the cluster giving a faster execution.

[11] https://grouplens.org/datasets/movielens/
[12] https://www.netflixprize.com/
[13] http://www.recsyschallenge.com/2017/

6 Acknowledgement

This work has been done in the scope of "PhD Seminar I" course. All sincere thanks to my professor Irene Rodrigues who helped me step by step to achieve this work. All thanks to the Global Platform for Syrian Students for all the support. All thanks to the SIAG company for all the support.

References

1. Dhruba Borthakur et al. Hdfs architecture guide. *Hadoop Apache Project*, 53, 2008.
2. John S Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc., 1998.
3. Min Chen, Shiwen Mao, and Yunhao Liu. Big data: A survey. *Mobile networks and applications*, 19(2):171–209, 2014.
4. Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016.
5. Jens Dittrich and Jorge-Arnulfo Quiané-Ruiz. Efficient big data processing in hadoop mapreduce. *Proceedings of the VLDB Endowment*, 5(12):2014–2015, 2012.
6. Robert Evans. Apache storm, a hands on tutorial. In *Cloud Engineering (IC2E), 2015 IEEE International Conference on*, pages 2–2. IEEE, 2015.
7. Zeno Gantner, Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Mymedialite: a free recommender system library. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 305–308. ACM, 2011.
8. Khairil Imran Bin Ghauth and Nor Aniza Abdullah. Building an e-learning recommender system using vector space model and good learners average rating. In *Advanced Learning Technologies, 2009. ICALT 2009. Ninth IEEE International Conference on*, pages 194–196. IEEE, 2009.
9. Trevor Hastie, Robert Tibshirani, and Jerome Friedman. Unsupervised learning. In *The elements of statistical learning*, pages 485–585. Springer, 2009.
10. James M Keller, Michael R Gray, and James A Givens. A fuzzy k-nearest neighbor algorithm. *IEEE transactions on systems, man, and cybernetics*, (4):580–585, 1985.
11. Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160:3–24, 2007.
12. Pieter M Kroonenberg and Jan De Leeuw. Principal component analysis of three-mode data by means of alternating least squares algorithms. *Psychometrika*, 45(1):69–97, 1980.
13. Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003.
14. Luis Martinez, Rosa M Rodriguez, and Macarena Espinilla. Reja: a georeferenced hybrid recommender system for restaurants. In *Web Intelligence and Intelligent Agent Technologies, 2009. WI-IAT'09. IEEE/WIC/ACM International Joint Conferences on*, volume 3, pages 187–190. IEEE, 2009.
15. Raymond J Mooney and Loriene Roy. Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 195–204. ACM, 2000.

16. Jyoti Nandimath, Ekata Banerjee, Ankur Patil, Pratima Kakade, Saumitra Vaidya, and Divyansh Chaturvedi. Big data analysis using apache hadoop. In *Information Reuse and Integration (IRI), 2013 IEEE 14th International Conference on*, pages 700–703. IEEE, 2013.
17. Carlos Porcel and Enrique Herrera-Viedma. Dealing with incomplete information in a fuzzy linguistic recommender system to disseminate information in university digital libraries. *Knowledge-Based Systems*, 23(1):32–39, 2010.
18. Gerard Salton, Anita Wong, and Chung-Shu Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.
19. Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.
20. Vinod Kumar Vavilapalli, Arun C Murthy, Chris Douglas, Sharad Agarwal, Mahadev Konar, Robert Evans, Thomas Graves, Jason Lowe, Hitesh Shah, Siddharth Seth, et al. Apache hadoop yarn: Yet another resource negotiator. In *Proceedings of the 4th annual Symposium on Cloud Computing*, page 5. ACM, 2013.
21. Maksims Volkovs, Guang Wei Yu, and Tomi Poutanen. Content-based neighbor models for cold start in recommender systems. In *Proceedings of the Recommender Systems Challenge 2017*, page 7. ACM, 2017.
22. Matei Zaharia, Reynold S Xin, Patrick Wendell, Tathagata Das, Michael Armbrust, Ankur Dave, Xiangrui Meng, Josh Rosen, Shivaram Venkataraman, Michael J Franklin, et al. Apache spark: a unified engine for big data processing. *Communications of the ACM*, 59(11):56–65, 2016.
23. Zhi-Dan Zhao and Ming-Sheng Shang. User-based collaborative-filtering recommendation algorithms on hadoop. In *Knowledge Discovery and Data Mining, 2010. WKDD'10. Third International Conference on*, pages 478–481. IEEE, 2010.

Sistemas de Recomendação para Grupos e Redes Neuronais*

Nuno Miranda
d11797@uevora.pt

Universidade de Évora, Departamento de Informática
Orientador: Teresa Gonçalves

Resumo Os sistemas de recomendação são uma área bastante rica em abordagens e algoritmos de diferentes tipos. Quando se pretende trabalhar nesta área, é fundamental um levantamento prévio dessas principais abordagens e aproximações. Esse levantamento de abordagens de estado da arte já tinha sido elaborado do ponto de vista mais teórico e também com a experimentação prática e comparação de resultados dos algoritmos mais utilizados. Neste trabalho pretendeu-se aprofundar o conhecimento das Redes Neuronais que é uma metodologia ainda pouco utilizada no contexto especifico dos algoritmos de recomendação para grupos. Este estudo tem o intuito fazer um levantamento de conceitos em Redes Neuronais e aplicar posteriormente essas técnicas nos sistemas de recomendação para grupos para assim obter-se métricas para análise e comparação dos diversos algoritmos já testados.

1 Introdução

Os sistemas de recomendação são técnicas e ferramentas de software que permitem sugerir a escolha de um, ou vários itens, ao utilizador [7,20]. As áreas onde estes sistemas são mais utilizados são na recomendação de compras, músicas, filmes, destinos de férias, notícias e livros.

Os sistemas de recomendação, especialmente aplicados no contexto web e das novas tecnologias, permitiram a resolução da limitação conhecida como *fenómeno de cauda longa*[1] [29,8], que é quando apenas os itens de maior sucesso são apresentados, renegando sempre para segundo plano os restantes, mesmo que sejam mais apelativos para o utilizador. A aplicação dos mecanismos de recomendação em contextos web veio alterar esse comportamento, pois os itens exibidos não estão restringidos aos mais populares. Graças aos sistemas de recomendação, é possível apresentar aos utilizadores itens pouco populares, mas que ainda assim, preenchem as preferências de certos utilizadores.

A designação de item ou itens é normalmente utilizada para designar o que vai ser recomendado pelo sistema, assim como o conjunto de onde será extraída essa recomendação [20]. O termo de utilizador é por norma atribuído à pessoa

* Este Artigo é destinado a Seminário IV
[1] Do Inglês 'Long Tail Phenomenon'

que vai usufruir da recomendação do sistema [20]. No entanto não são apenas os itens que são analisados pelos sistemas de recomendação. Dependendo do sistema e da abordagem seguida, o próprio utilizador também pode ser analisado durante esse processo [9,7,5]. As transações, são o termo frequentemente utilizado para designar as interações entre o utilizador e o sistema de recomendação. Interações essas, que fundamentalmente são a obtenção de dados que permitem aos algoritmos efetuar futuras recomendações [24].

A formalização do problema inerente aos mecanismos e sistemas de recomendação [1] passa por ter o utilizador c pertencente ao conjunto de utilizadores C e o item s pertencente ao conjunto de itens S com N elementos. Assim a função $U(c, s)$ é a responsável por obter a utilidade de uma recomendação. Para determinar o item ou itens com maior utilidade para um utilizador, é necessário aplicar essa função a todos os itens $U(c, s_1), ..., U(c, s_N)$. Após essa operação, podem-se obter os itens ordenados pela sua utilidade $s_{j_1}, ..., s_{j_N}$, ou apenas os K elementos mais relevantes $s_{j_1}, ..., s_{j_k}$ com $(K \leq N)$, ou ainda, apenas o item com maior utilidade, $s_j = \arg\max_{j \in S} U(c, j)$.

Para alcançar os itens com utilidade máxima existe um grande número de técnicas e de abordagens que são agrupadas em famílias por terem uma aproximação semelhante. Algumas utilizam técnicas de aprendizagem automática, teorias de aproximação e diversas heurísticas.

A função de utilidade nem sempre é obtida objetivamente para todos os itens, pois nem todos eles foram avaliados ou caracterizados de acordo com as preferências do utilizador. É sobretudo nesses itens não caracterizados que os sistemas de recomendação assumem uma importância elevada. Nesses casos a função de utilidade tem de ser estimada.

1.1 Abordagens de Filtragem de Conteúdos

Com esta abordagem, o sistema de recomendação gera os seus resultados de estimativa de utilidade, baseando-se nas características e atributos de outros itens que foram escolhidos ou preferidos pelo utilizador no passado. Ou seja, o cálculo da utilidade $U(c, s)$ para o utilizador c e para o item s é obtida através das utilidades dos outros itens para o mesmo utilizador $U(c, s_j)$, desde que os itens sejam todos semelhantes e com atributos comuns que permitam a comparação entre eles.

A aproximação baseada em conteúdos tem as suas origens em sistemas de extração de informação [3,23] e em sistemas de filtragem de informação [11]. Ambas as técnicas são bastante utilizadas porque muitos dos sistemas de recomendação trabalham sobre itens com informação descritiva em língua natural, como é o caso da recomendação de páginas web, de notícias e de livros.

No entanto, os melhores sistemas de recomendação baseados em conteúdos não se ficam pelas técnicas de extração e de filtragem de informação. Frequentemente adicionam técnicas de criação de perfis. Essas abordagens obtêm dados para além do item em si, recolhendo informação sobre as preferências do utilizador. Esses dados extra, como já foi introduzido anteriormente, podem ser obtidos de forma implícita [17,2] ou explícita [16,17].

Nos sistemas que assentam em itens baseados em textos em língua natural, recorre-se muitas vezes à computação e extração de palavras-chave. As palavras-chave obtidas dos diversos textos acabam por ser os atributos de comparação para a descoberta de itens que se enquadrem nos gostos do utilizador. Por exemplo o Sistema Fab [4] recomenda páginas Web ao utilizador tendo em conta as 100 palavras-chave mais importantes. De forma semelhante, o sistema Syskill & Webert [18] baseia as suas recomendações de páginas Web nas 128 palavras mais informativas. Para obter os termos mais significativos, existem várias aproximações, no entanto uma das mais utilizadas é o TFIDF[2] [11], que tem em conta a ocorrência do termo no documento em análise e no conjunto de todos os documentos de comparação do sistema de recomendação. Nesta técnica, um termo tem mais importância quanto mais se repete no documento, mas por sua vez, esse termo perde importância quantas mais vezes se repetir na coleção dos documentos.

1.2 Abordagens Colaborativas

Nos sistemas de recomendação baseados em abordagens colaborativas, a estimativa de utilidade sobre os diversos itens é efetuada a partir da análise das escolhas de outros utilizadores com perfil semelhante.

Uma das vantagens destas abordagens é permitir ter uma elevada, ou total, abstração sobre os itens e os seus atributos. Em situações onde os sistemas de recomendação baseados em conteúdos falham porque os itens têm poucos atributos ou esses atributos não são computáveis, esta limitação é completamente ultrapassada com os princípios das abordagens colaborativas [9,24]. Esta abordagem é uma das mais largamente utilizada na maioria das plataformas web que utilizam sistemas de recomendação com milhares ou milhões de itens e utilizadores, sendo até designada por correlação pessoa-pessoa [25].

Na construção de sistemas de recomendação deste tipo, e à semelhança dos sistemas de filtragem de conteúdos, também são criados vetores de atributos com pesos distintos para cada um deles, sendo posteriormente aplicadas diversas técnicas para correlacionar esses vetores. No entanto, neste caso, os atributos e respetivos valores são obtidos a partir de características dos utilizadores. A recolha dos atributos podem seguir uma aproximação explícita ou implícita (também à semelhança dos sistemas de filtragem de conteúdos).

Formalmente, esta abordagem baseia a sua recomendação na função utilidade para cada utilizador c e respetivo item s, sendo que a utilidade $U(c, s)$ é estimada através das utilidades $U(c_j, s)$, associadas ao item $s \in S$ e utilizadores $c_j \in C$ com características semelhantes a c.

O sistema Grundy [21] para a recomendação de livros, um dos primeiros a implementar esta abordagem, utilizava a designação de estereótipos para a aglutinação dos diferentes utilizadores em perfis, consoante as suas preferências. No entanto, a criação desses perfis era um processo inteiramente manual. Mais tarde,

[2] Sigla proveniente do Inglês, *Term Frequency-Inverse Document Frequency*

o sistema Tapestry [9], solicitava aos utilizadores que, manualmente, selecionassem outros utilizadores com gostos semelhantes, baseando-se no seu histórico de itens. Os sistemas GroupLens [14,19], Video Recommender [13] e Ringo [26] foram os primeiros a efetuar este tipo de operações automaticamente e a obter a respetiva recomendação automática. Mais tarde, surgiu uma nova geração de sistemas mais complexos e eficientes como o algoritmo de recomendação de livros da Amazon [15], o sistema PHOAKS [27] para a recomendação de informação relevante na Web e o sistema Jester [10] para a recomendação de anedotas online.

Os sistemas de recomendação colaborativos podem, segundo [6], ser agrupados em duas sub-classes tendo em consideração o seu funcionamento. A primeira classe, é baseada em memória ou heurísticas[3]; a segunda classe de algoritmos colaborativos é baseada em modelos[4].

2 Redes Neuronais

Após uma breve introdução dos conceitos principais envolvidos nos sistemas de recomendação, de seguida vai-se aprofundar o conhecimento em Redes Neuronais que é uma metodologia ainda pouco utilizada nos algoritmos de recomendação para grupos. Este estudo tem o intuito fazer um levantamento de conceitos em Redes Neuronais e aplicar essas técnicas em sistemas de recomendação para grupos para assim obter-se métricas para análise e comparação prática dos diversos algoritmos já testados em trabalhos anteriores.

2.1 Conceitos Base

Uma Rede Neuronal de uma forma muito simplista consiste numa estrutura interligada, na qual o processamento se encontra distribuído por um grande número de pequenas unidades densamente interligadas entre si [12,28]. Por analogia aos sistemas biológicos, estas unidades interligadas por onde o processamento é efetuado, são chamadas de neurónios.

Um neurónio biológico, representado na Figura 1 esquerda, consiste numa célula capaz de realizar um processamento simples. Cada neurónio é estimulado por uma ou mais ligações vindas de outros neurónios, chamadas sinapses. Dependendo do sinal produzido como da força das ligações, a sua natureza pode ser inibidora ou ativadora. Dependendo desses sinais de entrada, vai existir também um sinal de saída. Este sinal é propagado ao longo do axónio sendo propagado como sinal de entrada de outros neurónios. O funcionamento dos neurónios artificiais (Figura 1 direita) baseia-se numa analogia simplificada dos neurónios biológicos.

2.2 Perceptrão

A variante mais simples de implementar e de compreender, mas ainda assim bastante eficiente em problemas de classificação automática como classificador

[3] Do Inglês: *memory-based* ou *heuristic-based*
[4] Do Inglês: *model-based*

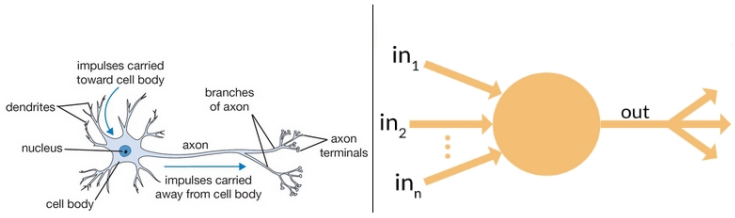


Figura 1. Analogia entre neurónio biológico e artificial

linear é o Perceptrão. O conceito teórico do Perceptrão foi criado por Frank Rosenblatt [22], tendo sido implementado em diversas linguagens, softwares e até em hardware específico.

No Perceptrão, como é visível na Figura 2, cada neurónio tem um conjunto de sinais de entrada $X = \{x_1, x_2, ..., x_n\}$ onde é ainda acrescentado um peso a cada variável de entrada $W = \{w_1, w_2, ..., w_n\}$. Esses pesos pretendem replicar o conceito biológico de sinapses com mais ou menos influência e se inibidoras ou ativadoras, dependendo do sinal das mesmas.

De seguida no Perceptrão temos a função soma, que a maioria das vezes apenas é o produto interno entre o vetor X e o vetor W e é a responsável por agregar todos os valores de entrada com os respetivos pesos e obter um único resultado.

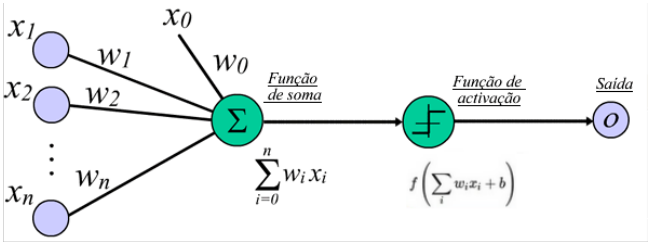


Figura 2. Esquema do Perceptrão

Por fim temos a função de ativação, que tem em conta o resultado da função soma e toma a decisão sobre o valor de saída do Perceptrão. A função de ativação pode tomar várias formas. A mais simplista é aplicando uma função binária em que o *threshold* é o valor definido como fronteira de ativação da função:

$$\text{Saída} = \begin{cases} 0 \text{ if } \sum_j w_j x_j \leq \text{ threshold} \\ 1 \text{ if } \sum_j w_j x_j > \text{ threshold} \end{cases}$$

No entanto, no contexto das redes neuronais, o *threshold* é passado para o lado esquerdo da equação e designado de *bias* sendo b = -*threshold* . Esta alteração

simples vai ser importante na fase de aprendizagem da rede neuronal. Assim a forma mais usual da função base de ativação toma a seguinte forma:

$$\text{Saída} = \begin{cases} 0 \text{ if } w \cdot x + b \leq 0 \\ 1 \text{ if } w \cdot x + b > 0 \end{cases}$$

2.3 Perceptrão sigmóide

O Perceptrão anteriormente descrito tem a limitação de ser demasiado drástico entre os seus valores binários de saída. Em certos cenários de aplicação isso não é problemático, no entanto, quando se pretende passar de um Perceptrão isolado para uma rede de Perceptrões (Rede Neuronal) onde se têm vários Perceptrões encadeados, esse comportamento binário iria tornar a rede neuronal numa simples árvore de decisão.

Para permitir que cadeias de Perceptrões sejam mais flexíveis e adaptáveis a cenários de classificação complexos, utilizam-se funções de ativação mais progressiva e não lineares, sendo a mais utilizada a função sigmóide. O facto de ser mais sensível permite ajustes durante a fase de aprendizagem numa rede neuronal que utilize esta função de ativação. Voltando à função de ativação sigmóide ela toma a seguinte forma:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Quando adaptada ao contexto dos Perceptrões com a adição dos sinais de entrada $X = \{x_1, x_2, ..., x_n\}$, os pesos $W = \{w_1, w_2, ..., w_n\}$ e o *bias* obtemos:

$$\frac{1}{1 + \exp(-\sum_j w_j x_j - b)}.$$

A representação gráfica da função sigmóide pode ser observada à esquerda na Figura 3. Ao contrário do Perceptrão, esta função não tem apenas os valores 0 e 1 como valores de saída, mas sim os números reais entre $[0, 1]$ e tem também um gradiente de transição suave entre esses valores de saída. Em certas redes neuronais, é interessante ter a função de ativação centrada em zero. Nesses casos utiliza-se a função tangente, que apresenta uma curva bastante semelhante à sigmóide mas com os valores de saída compreendidos entre $[-1, 1]$ como pode ser visualizado à direita da Figura 3.

2.4 Arquitetura de Redes Neuronais Artificiais

Uma Rede Neuronal ou Perceptrão Multi-camada não é mais que um conjunto de Perceptrões/neurónios conectados entre si num grafo acíclico, em que a saída de alguns desses neurónios está ligada à entrada de vários outros neurónios. Estão organizados na forma de um grafo acíclico para evitar ciclos infinitos. As redes neuronais não são grafos desorganizados, estão estruturados e agrupados em camadas. Por norma, essas camadas encontram-se sempre totalmente conectadas com as camadas adjacentes de neurónios. Quando se representam redes

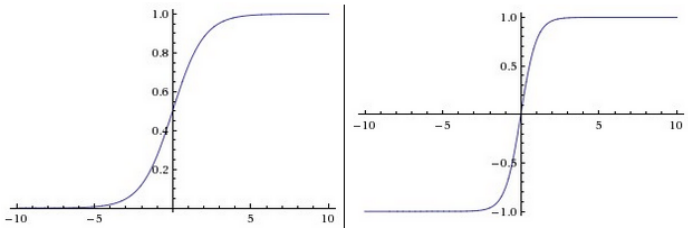


Figura 3. Função sigmóide e Tangente

neuronais, a camada de neurónios à esquerda é a camada de entrada, onde esses neurónios estão ligados a variáveis externas à rede. Já a camada mais à direita é a camada de saída que entrega o resultado final processado pela rede neuronal. As restantes camadas que ficam entre a camada de entrada e a camada de saída, chamam-se camadas escondidas[5]. Não existe nenhum motivo especial ou obscuro para se chamarem camadas escondidas, é simplesmente a designação dada às camadas que não são de entrada nem de saída.

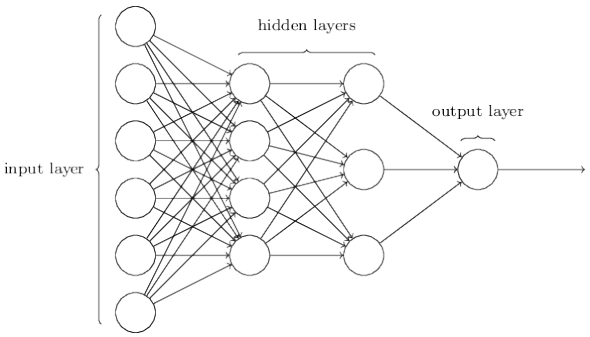


Figura 4. Rede Neuronal com 3 camadas

Na Figura 4 é possível observar um exemplo de Rede Neuronal com 3 camadas (a camada de entrada não é contabilizada). Para além da camada de entrada existem ainda duas camadas escondidas e a camada de saída. Já em relação às ligações/sinapses, todos os neurónios estão ligados com todos os neurónios das camadas adjacentes, no entanto não existe qualquer ligação entre neurónios da mesma camada. Na última camada, ou camada de saída, existe um neurónio por cada classe que se está a tentar classificar. Nesta última camada de neurónios não existe função de ativação. O valor de saída é apenas o valor referente à soma dos valores de entrada e respetivos pesos. Com um valor abaixo de 0,5 indica que

[5] Do Inglês: *Hidden Layers*

a classe não ocorre e a cima desse valor já ocorre, assumindo a função sigmóide nas camadas anteriores.

2.5 Contabilização de Elementos

Partindo dos exemplos da Figura 5 em que a vermelho está representada a camada de entrada, a azul as camadas escondidas e a verde as camadas de saída, temos as seguintes métricas.

- **Rede Neuronal Esquerda:** 6 neurónios (4 + 2) descartando os sinais de entrada, 20 pesos (3 x 4) + (4 x 2), 6 bias (4 + 2) e um total de 26 parâmetros ajustáveis.
- **Rede Neuronal Direita:** 9 neurónios (4 + 4 + 1) descartando os sinais de entrada, 32 pesos (3 x 4) + (4 x 4) + (4 x 1), 9 bias (4 + 4 + 1) e um total de 41 parâmetros ajustáveis.

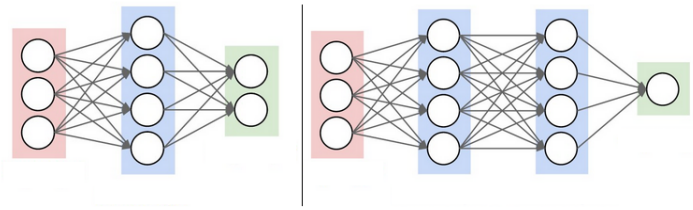


Figura 5. Contabilização de Elementos de uma Rede Neuronal

As redes neuronais que foram anteriormente contabilizadas são exemplos relativamente compactos. Em certos cenários complexos uma rede neuronal pode atingir cerca de 100 milhões de parâmetros e atingir profundidades até 20 camadas. Como o aumento de camadas implica um grande aumento no número de parâmetros, qualquer nova camada adicionada à Rede Neuronal aumenta fortemente a sua complexidade de cálculo e de utilização. Já o termo "Aprendizagem Profunda"[6] está relacionado com o número de camadas da Rede Neuronal.

2.6 Aprendizagem de uma Rede Neuronal

Uma Rede Neuronal para conseguir efetuar uma tarefa de classificação tem de ter a sua fase de aprendizagem onde é auto-ajustada à tarefa a realizar. Essa aprendizagem consiste em ajustar os pesos das variáveis de entrada de cada neurónio de modo a ajustar o valor de saída de toda a rede neuronal, e assim minimizar o erro entre o valor obtido e o valor esperado.

[6] Do Inglês: *Deep Learning*

A atualização dos pesos da rede neuronal é feita através da minimização do erro dado por uma função de custo, sendo a mais utilizada a do erro quadrático. Por sua vez os algoritmos de descida do gradiente são os mais usados no processo de minimização do erro.

O treino da rede pode ser feito usando a abordagem *Batch* ou a abordagem *Online*. Na primeira, cada atualização dos pesos é feita após a apresentação à rede de todos os casos do conjunto de treino. Já na segunda, a atualização dos pesos é feita após a apresentação de cada caso. Sempre que um conjunto de dados de entrada percorre toda a rede neuronal até à obtenção dos valores de saída, a esse percurso chama-se época. O número de épocas de treino pode ser fixo mas o mais apropriado é que seja orientado pelo erro no conjunto de validação. Como a rede neuronal na fase de aprendizagem vai-se ajustando e melhorando a cada iteração/época, o ideal é permitir que o sistema tenha o número suficiente de épocas até chegar a um ponto que não consiga minimizar mais o erro final.

2.7 Descida de Gradiente

A descida de gradiente, como anteriormente foi referido é uma otimização para minimizar o erro no valor de saída. Para perceber melhor esta otimização vamos primeiro trabalhar com apenas um Perceptrão.

O problema foca-se agora, em determinar a combinação de pesos (vetor de pesos) para o qual o perceptrão produz a saída correta para cada um dos exemplos de treino. Uma forma de encontrar esse vetor otimizado é começar por inicializa-lo com valores aleatórios. Depois, iterativamente aplicar o perceptrão a cada um dos exemplos e ajustar os pesos do vetor sempre que o perceptrão classifique erroneamente esse exemplo. O processo de treino termina quando a maioria dos exemplos de treino forem bem classificados pelo perceptrão.

Assim, cada peso w_i associado à entrada x_i é ajustado de acordo com a regra de treino do perceptrão obtendo-se assim o novo peso:

$$w_i' = w_i + \Delta w \quad \text{e} \quad \Delta w = \eta(t - o)x_i$$

Sendo t o valor pretendido e o o valor obtido pelo perceptrão. Já o η é uma constante de baixo valor, chamada de taxa de aprendizagem. Se o seu valor for muito elevado, faz com que o perceptrão se re-adapte inicialmente muito rápido, mas chegando a certo ponto deixa de convergir para minimizar o erro e toma valores caóticos no ajuste do perceptrão. Com valores baixos, a aprendizagem tem um comportamento mais suave e convergente. A regra anteriormente descrita apenas funciona bem em contextos de classificação linearmente separáveis. Em contextos mais complexos utiliza-se uma aproximação de forma a obter o melhor ajuste possível chamada de técnica de descida do gradiente que faz a procura, no espaço de hipóteses, dos vetores de pesos que melhor se ajustam aos exemplos de treino. Neste cenário o erro de treino já é obtido da seguinte forma:

$$E\left[\boldsymbol{w}\right] = \frac{1}{2}\sum_{d \in D}\left(t_d - o_d\right)^2$$

Onde D é o conjunto de exemplos a utilizar durante o processo de aprendizagem. Para demonstrar de uma forma visual o processo de descida de gradiente no espaço de hipóteses dos vetores de pesos e o respetivo erro E associado, pode ser visualizada a Figura 6. Sendo um exemplo simplista, apenas estão presentes dois pesos w_1 e w_2 nos eixos x e y, enquanto que o erro associado $E[w_1, w_2]$ encontra-se no eixo do z. Ainda no eixo z é aplicada a utilização de cores para facilitar a perceção visual e quantificação do erro. Sendo os tons quentes um erro maior, e os tons frios um erro menor. Na Figura 6 do lado esquerdo temos o momento inicial, com o erro obtido com os valores iniciais dos pesos w_1 w_2. Já na Figura 6 do lado direito, o Delta (Δ) já foi calculado e é possível visualizar a minimização do erro com a aplicação desse delta sobre os pesos iniciais. Isto retrata também o momento antes e depois de uma época na rede neuronal. De seguida iriam ocorrer diversas épocas e a cada iteração os pesos seria ajustados com novos Deltas e progressivamente o erro obtido iria ser minimizado.

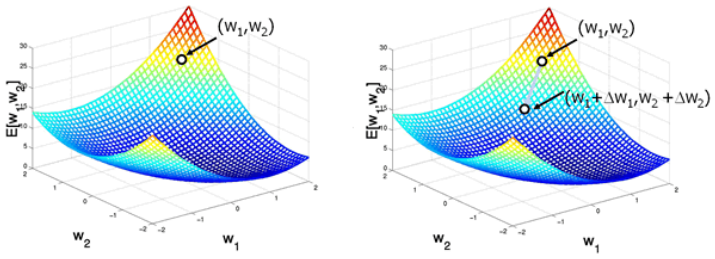


Figura 6. Demonstração da Descida de gradiente no espaço de hipóteses

Como o espaço de hipóteses pode ter tantas dimensões como os diversos pesos envolvidos, o gradiente geral é obtido pelo conjunto das diversas derivadas parciais, uma para cada dimensão/peso diferente, sendo o gradiente geral:

$$\nabla E[\boldsymbol{w}] = \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \ldots, \frac{\partial E}{\partial w_n}\right]$$

Que simplificando, o Delta para cada componente do peso fica:

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$$

3 Conclusões e Trabalho Futuro

Como conclusão deste levantamento de conceitos e metodologias dos sistemas de recomendação e de Redes Neuronais. Pode-se concluir que foram aprofundados

conhecimentos nessa área e assim solidificadas as bases de partida para o desenvolvimento de novas abordagens, especialmente na aplicação de Redes Neuronais em sistemas de recomendação para grupos.

Como trabalho futuro pretende-se criar uma nova abordagem experimental na área da recomendação para grupos recorrendo a Redes Neuronais e efetuar comparações com as abordagens existentes já experimentadas e assim criar novo conhecimento com uma metodologia ainda pouco utilizada nos algoritmos de recomendação para grupos.

Referências

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Trans. on Knowl. and Data Eng. 17(6), 734–749 (Jun 2005), http://dx.doi.org/10.1109/TKDE.2005.99
2. Anand, S.S., Mobasher, B.: Intelligent techniques for web personalization. In: Proceedings of the 2003 International Conference on Intelligent Techniques for Web Personalization. pp. 1–36. ITWP'03, Springer-Verlag, Berlin, Heidelberg (2005), http://dx.doi.org/10.1007/11577935_1
3. Baeza-Yates, R.A., Ribeiro-Neto, B.: Modern Information Retrieval. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1999)
4. Balabanović, M., Shoham, Y.: Fab: Content-based, collaborative recommendation. Commun. ACM 40(3), 66–72 (Mar 1997), http://doi.acm.org/10.1145/245108.245124
5. Basilico, J., Hofmann, T.: Unifying collaborative and content-based filtering. In: In ICML. pp. 65–72. ACM Press (2004)
6. Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence. pp. 43–52. UAI'98, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1998), http://dl.acm.org/citation.cfm?id=2074094.2074100
7. Burke, R.: The adaptive web. chap. Hybrid Web Recommender Systems, pp. 377–408. Springer-Verlag, Berlin, Heidelberg (2007), http://dl.acm.org/citation.cfm?id=1768197.1768211
8. Celma, Ò.: Music Recommendation and Discovery in the Long Tail. Ph.D. thesis, Universitat Pompeu Fabra, Barcelona (2008), static/media/PhD_ocelma.pdf
9. Goldberg, D., Nichols, D., Oki, B.M., Terry, D.: Using collaborative filtering to weave an information tapestry. Communications of the ACM 35, 61–70 (1992)
10. Goldberg, K., Roeder, T., Gupta, D., Perkins, C.: Eigentaste: A constant time collaborative filtering algorithm. Inf. Retr. 4(2), 133–151 (Jul 2001), http://dx.doi.org/10.1023/A:1011419012209
11. Hanani, U., Shapira, B., Shoval, P.: Information filtering: Overview of issues, research and systems. User Modeling and User-Adapted Interaction 11(3), 203–259 (Aug 2001), http://dx.doi.org/10.1023/A:1011196000674
12. Hertz, J., Krogh, A., Palmer, R.G.: Introduction to the Theory of Neural Computation. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1991)
13. Hill, W., Stead, L., Rosenstein, M., Furnas, G.: Recommending and evaluating choices in a virtual community of use. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 194–201. CHI '95, ACM

Press/Addison-Wesley Publishing Co., New York, NY, USA (1995), http://dx.doi.org/10.1145/223904.223929
14. Konstan, J.A., Miller, B.N., Maltz, D., Herlocker, J.L., Gordon, L.R., Riedl, J.: Grouplens: Applying collaborative filtering to usenet news. Commun. ACM 40(3), 77–87 (Mar 1997), http://doi.acm.org/10.1145/245108.245126
15. Linden, G., Smith, B., York, J.: Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing 7(1), 76–80 (Jan 2003), http://dx.doi.org/10.1109/MIC.2003.1167344
16. Mahmood, T., Ricci, F.: Improving recommender systems with adaptive conversational strategies. In: Proceedings of the 20th ACM Conference on Hypertext and Hypermedia. pp. 73–82. HT '09, ACM, New York, NY, USA (2009), http://doi.acm.org/10.1145/1557914.1557930
17. McSherry, F., Mironov, I.: Differentially private recommender systems: Building privacy into the net. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 627–636. KDD '09, ACM, New York, NY, USA (2009), http://doi.acm.org/10.1145/1557019.1557090
18. Pazzani, M., Billsus, D.: Learning and revising user profiles: The identification ofinteresting web sites. Mach. Learn. 27(3), 313–331 (Jun 1997), http://dx.doi.org/10.1023/A:1007369909943
19. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: Grouplens: An open architecture for collaborative filtering of netnews. In: Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work. pp. 175–186. CSCW '94, ACM, New York, NY, USA (1994), http://doi.acm.org/10.1145/192844.192905
20. Resnick, P., Varian, H.R.: Recommender systems. Commun. ACM 40(3), 56–58 (Mar 1997), http://doi.acm.org/10.1145/245108.245121
21. Rich, E.: Readings in intelligent user interfaces. chap. User Modeling via Stereotypes, pp. 329–342. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1998), http://dl.acm.org/citation.cfm?id=286013.286035
22. Rosenblatt, F.: Neurocomputing: Foundations of research. chap. The Perception: A Probabilistic Model for Information Storage and Organization in the Brain, pp. 89–114. MIT Press, Cambridge, MA, USA (1988), http://dl.acm.org/citation.cfm?id=65669.104386
23. Salton, G. (ed.): Automatic Text Processing. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1988)
24. Schafer, J.B., Frankowski, D., Herlocker, J., Sen, S.: The adaptive web. chap. Collaborative Filtering Recommender Systems, pp. 291–324. Springer-Verlag, Berlin, Heidelberg (2007), http://dl.acm.org/citation.cfm?id=1768197.1768208
25. Schafer, J.B., Konstan, J.A., Riedl, J.: E-commerce recommendation applications. Data Min. Knowl. Discov. 5(1-2), 115–153 (Jan 2001), http://dx.doi.org/10.1023/A:1009804230409
26. Shardanand, U., Maes, P.: Social information filtering: Algorithms for automating word of mouth. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 210–217. CHI '95, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA (1995), http://dx.doi.org/10.1145/223904.223931
27. Terveen, L., Hill, W., Amento, B., McDonald, D., Creter, J.: Phoaks: A system for sharing recommendations. Commun. ACM 40(3), 59–62 (Mar 1997), http://doi.acm.org/10.1145/245108.245122
28. Wasserman, P.D.: Neural Computing: Theory and Practice. Van Nostrand Reinhold Co., New York, NY, USA (1989)

29. Yin, H., Cui, B., Li, J., Yao, J., Chen, C.: Challenging the long tail recommendation. CoRR abs/1205.6700 (2012), http://arxiv.org/abs/1205.6700

Named-Entity Recognition for Portuguese Police Reports

Gonçalo Carnaz[1,2], Vitor Beires Nogueira[1,2], Mário Antunes[3,4], and N.M Fonseca Ferreira[5,6,7]

[1] Informatics Departament, University of Évora, Portugal
[2] LISP - Laboratory of Informatics, Systems and Parallelism, Portugal
[3] School of Technology and Management, Polytechnic Institute of Leiria, Portugal
[4] INESC-TEC, CRACS, University of Porto, Portugal
[5] Institute of Engineering of Coimbra, Polytechnic Institute of Coimbra, Portugal
[6] Knowledge Research Group on Intelligent Engineering and Computing for Advanced Innovation and Development (GECAD) of the Institute of Engineering, Polytechnic Institute of Porto, Portugal
[7] INESC TEC, Portugal

Abstract. During a criminal investigation several text documents are produced by police officers, creating a deluge of unstructured data obtained from heterogeneous sources. Therefore, identification and recognition of entities, i.e. places, organizations or persons, by a natural language pipeline, with named-entities recognition task, could help police officers to understand and find relevant information in data extracted. We aim to defined a natural language processing pipeline to identify and recognize entities from these police reports, supported by two trained corpus, namely Amazonia and a Portuguese News Corpus. Additionally, we evaluate named-entities recognition systems, focus in Portuguese language, with a dataset produced by the Portuguese police. We then evaluate the performance obtained on the information retrieval process applied to the dataset.

Keywords: natural language processing, named entity recognition, criminal investigation, police reports

1 Introduction

Criminal police must deal with a huge quantity of data acquired daily, or produced during investigations, from heterogeneous sources, like paper documents, digital reports, handwritten transcripts of interrogations, social media messages, transcripts or forensic logs. In every investigation, a final report is produced and includes analysis from different sources, made by texts and images. Therefore,

we have unstructured data to be processed by natural language processing systems, through a procedure designated named-entity recognition (described in section 2).

The necessity of understanding and manipulation of texts and speech, produced by humans, determines Natural Language Processing (NLP) as computer science field applicable to our research. It is a huge challenge for this computer science field, sought by several computer scientists. Therefore, the NLP arises as the solution for that challenge, supported by other fields, e.g. linguistics, mathematics, artificial intelligence, robotics, psychology and others. By definition, Natural Language Processing (NLP) is defined as,

> *"...a field of computer science,artificial intelligence,and linguistics concerned with the interactions between computers and human (natural) languages. Many challenges in NLP involve natural language understanding, that is, enabling computers to derive meaning from human or natural language input, and others involve natural language generation."* [1] .

NLP is build under different tasks, such as sentence detection and tokenization [11], stemming [13], Part-of-Speech Tagging [1], named-entities recognition (NER) [20] [17], relation extraction and others.

The rest of the article is organized as follows: In section 2 we present an introduction to natural language processing and named entities recognition; in section 3 described related work about natural language processing (NLP) and named entities recognition related to crime domain. Additionally, we presented a review of related works for Portuguese language from different domains. In section 4 we defined our setup environment related to the selected frameworks and the performance measures obtained; in section 5 present our NLP with results obtained with two trained Corpus, which are Amazonia and Portuguese News. The paper ends with conclusion and future work in section 6.

2 Named-entity recognition

The Sixth Message Understanding Conference [8] (MUC-6), introduced the *Named-Entity Recognition* task, as an activity to extract terms related to different entities, i.e. persons, cities, date and time or other entities extracted from structured and unstructured documents. It was defined as a sub-task of information extraction [17]. In [17] authors defined NER as

> *"...is a sub problem of information extraction and involves processing structured and unstructured documents and identifying expressions that refer to peoples, places, organizations and companies. For us, humans,*

[8] http://www.itl.nist.gov/iaui/894.02/related_projects/muc/

> *NER (Named-Entity Recognition) is intuitively simple, because many named entities are proper names and most of them have initial capital letters and can easily be recognized by that way, but for machine, it is so hard. One might think the named entities can be classified easily using dictionaries, because most of named entities are proper nouns, but this is a wrong opinion. As time passes, new proper nouns are created continuously".*

Along years several approaches were made from the language factor, textual genre or domain factor to Entity type factor [17].

2.1 Information retrieval metrics

Information retrieval field defined metrics to measure entity recognition extraction systems performance. The metrics [14] established are: P : Precision, R : Recall and $F - Measure$.

Precision is defined by the ratio of correct answers (True Positives) among the total answers produced (Positives),

$$P(\ Precision) = \frac{TP}{TP + FP}$$

where TP - *True Positive*, a predicted value was positive and the actual value was positive and FP - *False Positive*, predicted value was positive and the actual value was negative [12].

R - Recall is defined as a ratio of correct answers (True Positives) among the total possible correct answers (True Positives and False Negatives),

$$R(\ Recall) = \frac{TP}{TP + FN}$$

where FN - *False Negative*, a predicted value was negative and the actual value was positive [12].

$F - Measure$ - is a harmonic mean of precision and recall,

$$F\text{-}Measure = \frac{2 * precision * recall}{precision + recall}$$

3 Natural language processing system applied to crime domain - related work

In this section we describe named-entities recognition tasks and how these systems detect entities, i.e. places, persons, organizations or entities related to crime, from different languages. Additionally, we also describe systems developed for Portuguese language.

In 2010, [18] authors proposed an information extraction architecture to provide the input to a web-based system called WikiCrimes [10]. To analyze the extracted texts, they use a module called MorphoSyntactic Parser that performs a morphological and syntactic analysis creating a syntactic tree. In 2012, authors [24] proposed a system to extract Arabic named entities from crimes documents. The system used a standard preprocessing phrase, using a sentences splitting, tokenizer, Part-of-speech (POS) tagging (using a supervised statistical algorithm, trained with a corpus of crime related documents with 19800 words, in Arabic language) and a noun phrase chuncker. Follow by, a Named-Entity Identification and classification phrase with a Named-Entity Extraction, using a gazetteer (with a lexicon constituted by terms, i.e. Person, Personal properties, Location, Organizations and Indicative Words - crime terms), and a Pattern Rules module to train the NER to tag crime entities in documents.

In 2014, [8] proposed a NER system to extract entities from legal documents, e.g. judges, companies, courts or others. In [3] is proposed a system to extract crime information from online newspapers is proposed, fucused in the "hidden" information related to the theft crime.

In 2015 authors proposed crime information extraction from the Web, with crime NER task, using classification algorithms, e.g. Naive Bayes, Support Vector Machine and K-Nearest Neighbor. These classification algorithms are used to features extraction, through a voting combination module for features identification. Alongside, a indexing module that aims crime type identification, using the same classification algorithms [23] . In [25] authors proposed an approach based on raw text and extracts semi-structured information, in automatic way, using text mining techniques.

In 2016 authors proposed a system to extract verbs and their use, from crime clusters, using two data-sets, namely real data-sets from crime and industrial datasets with benchmarks. This system was defined with different tasks, removing not relating information, a stop words task with 571 words to delete from processed documents. Additional, the Porter stemmer for word stemming. For verbs identification, authors used a Word-Net identification method using the two datasets enumerated above [5].

Authors proposed in 2017 a named-entity recognition system for police documents for Dutch Police. The NER system is named *Frog*, using a traditional classification and evolution paradigm. With an annotated corpus, created from 250 criminal complaints reports, where domain experts identified entities, like location, person, organization, event, product and others [22]. In [2] methods are applied to discover criminal communities, analyzing their relations, and extract useful information from criminal text data.

There are several works related to natural language processing in native Portuguese language from different domains. Therefore, CaGE system [9] proposed

a recognition and disambiguation system of geographic named entities mapping with geographic information, e.g. latitude and longitude coordinates. The main features are to identify and to disambiguate geographic entities, on a dictionary and a geographic dictionary. PorTexTO [7] proposed a system for named-entities processing, related to time. This system was created for HAREM [21] evaluation campaign. R3M [16] developed a NER system to identify and classify entities, e.g. people, organizations and locations. It is based on a semi-supervised learning approach rather them linguistic resources. Rembrant system [15] proposed a NER and relation detection between named entities in Portuguese texts, with source of knowledge the Wikipedia. SEI-Geo [6] is a NER system for identification and classification of named entities, e.g. Locations, based on geo-ontologies and patterns.

4 Experimental setup

In the following paragraphs, we will describe the setup procedure for frameworks evaluation, the dataset and the frameworks used. Finally, a discussion about obtained results, following the information retrieval metrics (explain in section 2.1).

4.1 Setup procedure

We have designed a setup procedure, see picture 1, to explain the steps taken to obtain the metrics (precision, recall and F-measure) from the frameworks analyzed. In step one, we use as input a dataset (see section 4.2) created from a police report, that is a final report with documents elaborated during investigations, e.g., forensic reports or smartphone logs. The original police report, in MS Word format, was parsed to plain text (raw text), used in the NLP pipeline.

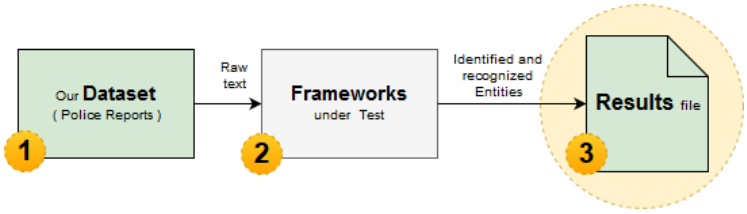


Fig. 1. *Test procedure workflow*

In step two, the text extracted will be processed by NER modules on each selected framework. Finally, the frameworks outputs are assessed. We will determine the *FN* (False Negatives), *FP* (False Positive) and *TP* (True Positives).

4.2 Dataset

We used a police report as a dataset, provided by a Portuguese police department and created during a drug crime investigation. the original file is in Microsoft Word format, with the following properties:

- Word count: 4657;
- Character count: 25152;
- Line count: 209;
- Paragraph count: 59;
- Other: tables and images.

The original file was processed, by a piece of code (supported by TIKA [9]), that parses the file into plain text (raw text), used as NLP pipeline input.

The focus of our experiment is to detect named entities, e.g. Person (PER), Organization (ORG), Locations (LOC) and Date (DAT) and how the selected frameworks process the dataset. Our dataset was annotated by domain specialist, using the following rules:

- Person (PER): identify persons names with a minimum of two words, i.e. politicians, scientists, artists or athletes;
- Organization (ORG): identified by full name or abbreviation, i.e. newspapers, banks, universities, schools, non-profits, companies or public services;
- Locations (LOC): identify by full address's or locals, i.e. countries, streets, cities or village's
- Date (DAT): identify by different formats, i.e. April 13 or 12/03/2013.

Entities extracted from our annotation procedure produce the following values: Person (PER) - 202; Organization (ORG) - 11; Locations (LOC) - 46 and Date (DAT) - 26.

4.3 Frameworks selected

In section 3, we described different approaches to NLP with NER tasks. The evaluation of these approaches could determined how our dataset will be processed by them, and what entities are identified and classified. We select approaches that are open source or trial versions for Portuguese or English language. Our focus is to identify named-entities in Portuguese, but in certain cases and because the framework was developed and trained for another language, we still evaluate them for discarding option. The selected frameworks are:

[9] https://tika.apache.org/

– KNIME (R) Analytics Platform [10]: is an open solution for data-driven innovation, that supports data mining and predicting. Among the predefined workflows in KNIME (R) Analytics Platform there is one for Natural Language Processing that is based on OpenNLP [11] and includes the process of Named-Entity Recognition;
– Linguakit [12]: created by the ProLNat@GE Group [13] (CITIUS, University of Santiago de Compostela) as a multilingual toolkit for NLP;
– RAPPort - A Portuguese Question-Answering System [19]: authors proposed a question answering system, supported by indices that store triples, related sentences and documents, using a NLP pipeline. This system is using CHAVE Corpus [14];

Each approach has somehow NLP toolkits that allow the development of the underlying tasks in an NLP pipeline, e.g. NLTK [15], Stanford CoreNLP [16], Pattern [17] and Polyglot [18].

4.4 Comparison results

To measure the performance of enumerated frameworks a set of metrics were used, i.e. Precision (P), Recall (R) and F-Measure (F1). The table 1 shows the performance measures:

Table 1. Frameworks performance metrics

	PER			**ORG**			**LOC**			**DAT**		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
Knime (NLP Workflow)	53%	18%	13%	30%	30%	10%	5%	5%	3%	-	-	-
RAPPort (DEI-UC)	51%	59%	55%	8%	50%	13%	48%	58%	53%	98%	87%	92%
Linguakit	67%	28%	39%	12%	82%	21%	50%	78%	60%	90%	90%	90%

Globally, the RAPPort approach reached the highest F-measure result for each entities (approximately 55%, 53% and 92%, respectively for Organization, Location and Date entities) for the detected entities, having the best trade-off

[10] https://www.knime.com
[11] opennlp.apache.org/
[12] https://github.com/citiususc/Linguakit
[13] http://gramatica.usc.es/pln/
[14] http://web.letras.up.pt/traducao/index_files/Page2219.htm
[15] www.nltk.org
[16] stanfordnlp.github.io/CoreNLP/
[17] www.clips.ua.ac.be/pattern
[18] polyglot.readthedocs.io/

regarding both measures (precision and recall). The best result obtained, related to F-measure for Organization entity was obtain by Linguakit, giving the best trade-off between precision and recall.

5 Our NLP Pipeline proposal

Our NLP pipeline proposal, is based on RAPPort [19], supported by three phases, that follows a standard NLP pipeline. In the first phase, we defined the data source and a parser module, the output of this phase is the processed original file (police report) into raw text, an important feature is data cleaning, e.g. removing formatting, images and tables. In the second phase, a pre-processing pipeline with a sentence boundary, tokenizer, stemming and POS Tagging tasks will prepare data for the next phase, the named-entities recognition module.

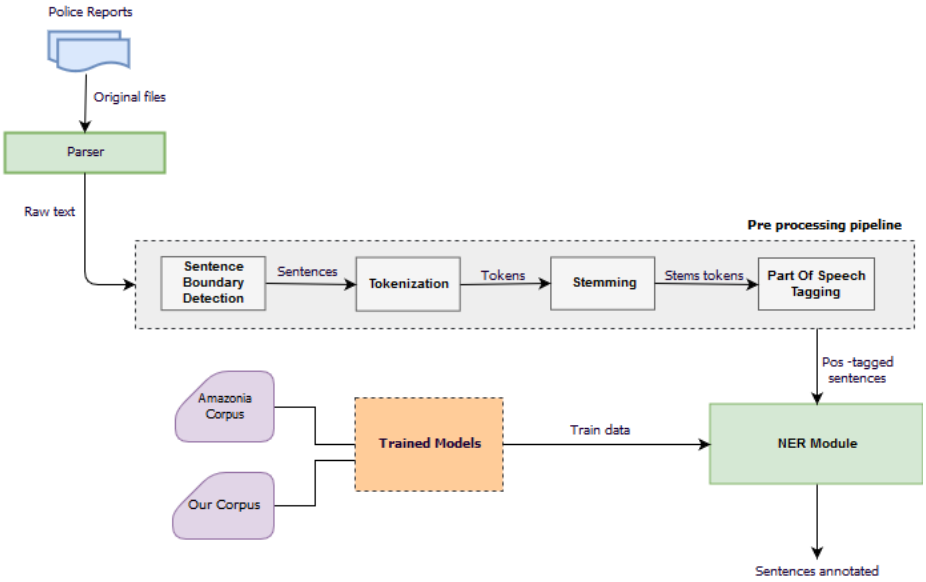


Fig. 2. *Proposed NLP Pipeline*

To complete these framework, we have a NER module supported by a trained corpus. We trained our model for named-entity recognition with two different corpus, e.g. Amazonia Corpus and Our News Crime Corpus. First, the Amazonia Corpus [19] has 4.6 millions of words (about thousand sentences) retrieved from Overmundo website, in Portuguese - Brazilian language, annotated by PALAVRAS [4]. The table 2 describes the trained corpus and a result of OpenNLP tool for training models:

[19] https://www.linguateca.pt/Floresta/corpus.html

Table 2. Amazonia Corpus data summary

Amazonia Corpus	
Sentences	81049
Tokens	1542622
Named-Entities	
Person	25237
Time	5490
Organization	20523
Place	15612

Regarding the second corpus, our motivation was to create a new corpus from portuguese online news about crime, e.g. Publico [20], Diário de Noticias [21] or Diário de Coimbra [22]. According to the domain experts the syntax and semantics of these news are similar to police language presented in the police reports. After that, we trained the corpus with OpenNLP [23] training tool, with the results described in table 3:

Table 3. Our News Corpus data summary

PT News Crime Corpus	
Sentences	310
Tokens	12078
Named-Entities	
Person	40
Time	57
Organization	82
Place	45

5.1 Results obtained

Following the setup procedure, we evaluated our dataset with our proposal for each corpus, obtaining the results described in table 4:

[20] https://www.publico.pt/
[21] https://www.dn.pt/
[22] http://www.diariocoimbra.pt/
[23] https://opennlp.apache.org/

Table 4. Evaluation results

	PER			ORG			LOC			DAT		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
Amazonia Corpus	66%	79%	72%	6%	67%	11%	27%	42%	33%	33%	92%	48%
Our Corpus	-	-	-	68%	13%	22%	97%	29%	44%	73%	56%	64%

Globally, with Our Corpus we reached the highest F-measure result for each entities (approximately 22%, 44% and 64%, respectively for Organization, Location and Date entities) for the detected entities, having the best trade-off regarding both measures (precision and recall). There is an entity that was not detected using Our Corpus, the Person entity, the reason for this failure detection is because our corpus does not have sufficient data for a fine train.

6 Conclusion and future work

The work developed was focused on the evaluation of open source frameworks for named-entity recognition retrieved from unstructured data, and a framework proposal for named-entity recognition with two trained corpus. In both cases, performance measures were performed, but other conclusions and investigation paths will be considered. We have obtained promising results with the frameworks analyzed, in almost all entities.. But our focus is the crime domain, therefore the obtained results were weak, no entity related to crime domain was identified. There are approaches, described in section 3, that identify and recognize entities related to crime for English or other languages. Our NLP framework proposal for named-entity recognition retrieved from Portuguese police reports, tries to increase named-entities detection adding two trained corpus, supporting police reports parsing to a common format. The preliminary results encourage the approach taken, but with improvements to be realized, e.g. a better trained corpus or identify and recognize entities related to crime.

Future work will consist of the framework improvement, clarifying the possibility of extracting named entities and relations from police reports, recognizing the entities related to crime, e.g. crime or narcotics. Additionally, we plan to increase our corpus quality to improve performance measures of our framework proposal.

References

1. N. Adhvaryu and P. Balani. Survey : Part-Of-Speech Tagging in NLP. *International Journal of Research in Advent Technology*, 1(1):102–107, 2015.

2. R. Al-Zaidy, B. C. M. Fung, and A. M. Youssef. Towards discovering criminal communities from textual data. In *Proceedings of the 2011 ACM Symposium on Applied Computing*, SAC '11, pages 172–177, New York, NY, USA, 2011. ACM.
3. R. Arulanandam, B. T. R. Savarimuthu, and M. A. Purvis. Extracting crime information from online newspaper articles. In *Proceedings of the Second Australasian Web Conference - Volume 155*, AWC '14, pages 31–38, Darlinghurst, Australia, Australia, 2014. Australian Computer Society, Inc.
4. E. Bick. *THE PARSING SYSTEM "PALAVRAS" Automatic Grammatical Analysis of Portuguese in a Constraint Grammar Framework "*. PhD thesis, University of Århus, 2000.
5. Q. Bsoul, J. Salim, and L. Q. Zakaria. Effect Verb Extraction on Crime Traditional Cluster. *World Appl. Sci. J.*, 34(9):1183–1189, 2016.
6. M. S. Chaves. Geo-ontologias e padrões para reconhecimento de locais e de suas relações em textos: o SEI-Geo no Segundo HAREM. In *Desafios na avaliação conjunta do reconhecimento entidades mencionadas O Segundo HAREM*, pages 231–245. 2008.
7. O. Craveiro. PorTexTO: sistema de anotação/extracção de expressões temporais. In *Desafios na avaliação conjunta do reconhecimento entidades mencionadas O Segundo HAREM*, pages 159–170. 2008.
8. C. Dozier, R. Kondadadi, M. Light, A. Vachher, S. Veeramachaneni, and R. Wudali. Semantic processing of legal texts. In E. Francesconi, S. Montemagni, W. Peters, and D. Tiscornia, editors, *Semantic Processing of Legal Texts*, chapter Named Entity Recognition and Resolution in Legal Text, pages 27–43. Springer-Verlag, Berlin, Heidelberg, 2010.
9. B. Emanuel. *Geographically Aware Web Text Mining*. PhD thesis, Universidade de Lisboa, 2008.
10. V. Furtado, L. Ayres, M. D. Oliveira, E. Vasconcelos, C. Caminha, J. D. Orleans, and M. Belchior. Collective intelligence in law enforcement – The WikiCrimes system. *Inf. Sci. (Ny).*, 180(1):4–17, 2010.
11. V. Gupta, L. C. Science, and G. S. Lehal. A Survey of Text Mining Techniques and Applications. *J. Emerg. Technol. Web Intell.*, 1(1):60–76, 2009.
12. I. M. Konkol. *Named Entity Recognition*. PhD thesis, University of West Bohemia, 2015.
13. B. Lovins. Development of a Stemming Algorithm. *Mech. Transl. Comput. Linguist.*, 11(June):22–31, 1968.
14. A. Mansouri, L. S. Affendey, and A. Mamat. Named Entity Recognition Approaches. *Journal of Computer Science*, 8(2):339–344, 2008.
15. E. Mencionadas. REMBRANDT - Reconhecimento de Entidades Mencionadas Baseado em Relações e ANálise Detalhada do Texto. In *Desafios na avaliação conjunta do reconhecimento entidades mencionadas O Segundo HAREM*, pages 195–211. 2008.
16. C. Mota. R3M, uma participação minimalista no Segundo HAREM. In *Desafios na avaliação conjunta do reconhecimento entidades mencionadas O Segundo HAREM*, pages 181–193. 2008.
17. D. Nadeau and S. Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

18. V. Pinheiro, V. Furtado, T. Pequeno, and D. Nogueira. Natural language processing based on semantic inferentialism for extracting crime information from text. In *2010 IEEE International Conference on Intelligence and Security Informatics*, pages 19–24, May 2010.
19. R. Rodrigues and P. Gomes. Rapport- a portuguese question-answering system. In F. Pereira, P. Machado, E. Costa, and A. Cardoso, editors, *Progress in Artificial Intelligence*, pages 771–782, Cham, 2015. Springer International Publishing.
20. C. J. Saju and A. S. Shaja. A survey on efficient extraction of named entities from new domains using big data analytics. In *2017 Second International Conference on Recent Trends and Challenges in Computational Models (ICRTCCM)*, pages 170–175, Feb 2017.
21. D. Santos, N. Seco, N. Cardoso, and R. Vilela. HAREM : An Advanced NER Evaluation Contest for Portuguese.
22. M. Schraagen. Evaluation of Named Entity Recognition in Dutch online criminal complaints. *Comput. Linguist. Netherlands J.*, 7:3–15, 2017.
23. H. A. Shabat and N. Omar. Named Entity Recognition in Crime News Documents Using Classifiers Combination. *Middle-East J. Sci. Res.*, 23(6):1215–1221, 2015.
24. A. I. Technology, M. Asharef, N. Omar, and M. Albared. Arabic Named Entity Recognition in Crime. *J. Theor. Appl. Inf. Technol.*, 44(1):1–6, 2012.
25. Y. Yang, M. Manoharan, and K. S. Barber. Modelling and analysis of identity threat behaviors through text mining of identity theft stories. In *2014 IEEE Joint Intelligence and Security Informatics Conference*, pages 184–191, Sept 2014.

Detecting Tuberculosis Multi-drug Resistance and Tuberculosis type

Md Sajib Ahmed, Md Obaidullah Sk, Teresa Gonçalves, and Luís Rato

[1] Department of Informatics, University of Évora, Portugal
[2] {jack6148,sk.obaidullah}@gmail.com, {tcg,lmr}@uevora.pt

Abstract. Tuberculosis is one of the most ancient germicidal infections caused by a microbe called Mycobacterium tuberculosis. Specifically, the distinction between drug-sensitive and multidrug-resistant tuberculosis is still problematic, and it's also difficult to detect the tuberculosis type. With this in mind, machine learning models able to determine if a patient suffers from multidrug-resistant tuberculosis and identify the type of tuberculosis based Computed Tomography (CT) scans of patients' lungs were developed. Specifically, texture analysis was used to generate features' values from CT scans and different types of classifiers were tested. An accuracy of 62.87% and a ROC Area (AUC) of 0.66 were obtained for the multidrug-resistant tuberculosis task and a 39.0% accuracy and 0.222 Kappa statistics for tuberculosis type detection.

Keywords: Tuberculosis, Computed Tomography, Texture Analysis, Classification

1 Introduction

Tuberculosis (TB) is an infection caused by a bacteria named Mycobacterium tuberculosis. These bacteria generally affect the lungs, but sometimes they can damage other parts of the body. TB spreads through the air when a person with lungs or throat TB coughs, sneezes, or talks. From World Health Organization (WHO) report, tuberculosis is one of the top 10 courses of death worldwide [17]. The greatest disaster that can happen to a patient with TB is that the organisms become resistant to two or more of the standard drugs. In contrast to drug sensitive (DS) TB, its multidrug resistant (MDR) form is much more difficult and expensive to recover from. Thus, early identification of the drug resistance (DR) status is of great importance for an effective treatment. The most frequent used methods of DR detection are either costly or take too much time (up to several months), therefore there is a need for quick and at the same time cheap methods of DR detection. One of the possible approaches for identifies the drug resistance and detects the type of tuberculosis to analysis the Computed Tomography (CT) image.

ImageCLEF (the image retrieval and analysis evaluation campaign of the Cross Language Evaluation Forum, CLEF) has organized challenges on image classification and retrieval since 2003 [16]. Since 2004, a medical image analysis

and retrieval task has been organized [12]. Our work intent to deliver system for ImageCLEFtuberculosis 2017 tasks [7].

The rest of the paper is organized as follows: Section 2 describes the Literature Review; Methodology is explained in Section 3, followed the presentation of Experiments (dataset description, evaluation metrics, system configuration, results and discussion) on Section 4; finally, Section 5 concludes the paper.

2 Literature Review

Early appropriate identification of the presence of drug resistance (MDR) and accurate diagnosis of TB type can reduce the potential detrimental effects on patients. Significant research has been done in TB disease field to accurately identify the drug resistance and detects the type of tuberculosis. Following described system were submitted to ImageCLEFtuberculosis 2017.

Braun *et al.* [3], proposed a system for multidrug-resistant tuberculosis (MDR TB) that is difficult to distinguish from drug-sensitive tuberculosis (DS TB). In the proposed system, CT scans images are pre-processed using the nibabel python library, then the three dimensional data sets are used in a three dimensional convolutional neural network (based on KERAS with TensorFlow as backend). They get 56.81% accuracy and 0.58 AUC using their system.

Cid *et al.* [4] present a graph-model of the lungs capable of characterizing TB patients with different lung problems. This graph-model contains a fixed number of modes with weighted edges based on distance measures between texture descriptors computed on the nodes. This model attempts to encode the texture distribution along the lungs, making it suitable for describing patients with different tuberculosis types. Using their model, they get 58.25% accuracy and 0.5164 AUC to identify the drug resistance, and 40.33% accuracy and 0.244 kappa statistics to detect the type of tuberculosis.

On another work, Cid *et al.* [6] used a non-parametric approach for characterizing heterogeneous diseases in large-scale studies. This is applied on CT images of Chronic Obstructive Pulmonary Disease (COPD) patients; it consists of describing each subject as a collection of local feature descriptors embedded in a dissimilarity space. The set of local features was extended for their work adding new 3D texture descriptors making this approach able to characterize several TB types, but not suitable for predicting multidrug resistance.

Sun *et al.* [22] describe a system that improves the diagnosis accuracy of drug resistant tuberculosis and also the identification of the type of tuberculosis present in the patient. First they use Convolutional Neural Networks (CNN), which are able to identify useful features in the Computerized Tomography (CT) scans, and perform the classification based on them. Then Recurrent Neural Networks (RNN) are used on top of CNNs by utilizing CNNs as a feature extractor and the RNNs as a classifier.

Liauchuk and Kovalev [14] proposed a method based on co-occurrence of adjacent super voxels in 3D CT images to distinguishing between multidrug resis-

tant tuberculosis (MDR TB) cases and drug sensitive (DS) ones and classifying the tuberculosis type.

For the same tasks, Silva *et al.* [19] propose a two-stage pipeline: data pre-processing and a Deep Learning (DL) model. Their pre-processing stage use the Computed Tomography (TB) images, segmenting the lungs and resizing data to be ready to feed the DL model. On the other hand, the DL model uses batches of pre-processed data for classification.

Stefan *et al.* [20] used a deep convolutional neural network architecture co-denamed Inception for the same tasks. The main hallmark of this architecture is the improved utilization of the computing resources inside the network.

3 Methodology

The block diagram of the proposed system (used for both problems at hands) is presented in Figure 1. First the input data set is pre-processed: slice extraction, ROI generation using a mask and ROIs selection based on threshold values are done; then, using Texture analysis on each ROI features are extracted a slice-wise averaging attribute values are calculated; finally, a feature vector is computed, followed by classification and performance analysis of multiple classifiers. The main parts of the system are described below.

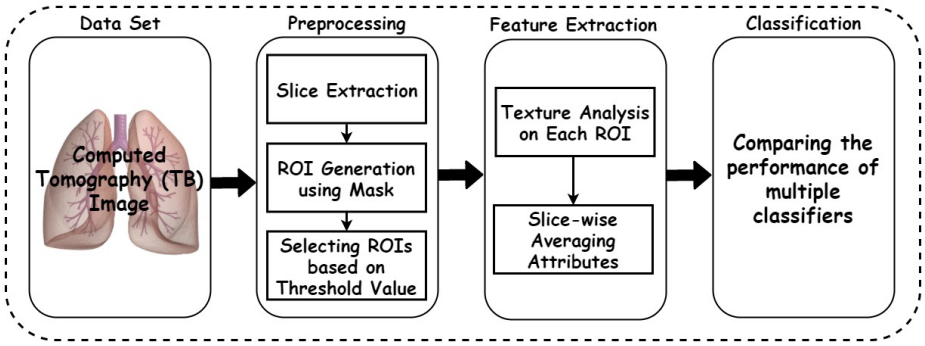


Fig. 1. Overview of the proposed system

3.1 Pre-processing

Slice Extraction, ROI Generation using Mask and ROI selection based on Threshold values are the parts of the pre-processing step and are detailed next.

Slice Extraction. Since the dataset images were in NIFTI format (where a single image has several slices), slices needed to be extracted from the 3D CT scan images. Figure 2 shows a slice of Tuberculosis CT scan image.

ROI Generation using mask. After slice extraction of each CT scan 3D images, a ROI for each slice was defined based on the given provided mask. Figure 3 shows a mask for image depicted on Figure 2

Fig. 2. A slice of Tuberculosis CT scan

Fig. 3. Provided Mask image of the lung

ROI selection based on threshold value. Observing each slice pattern, a threshold value of 15000 was chosen to ensure that no slices with meaningful information would be missed. Here, meaningful information means that some dots are present in the ROI.

3.2 Feature Extraction.

There are two parts for the feature extraction task: Texture Analysis on each ROI and Averaging Attribute Value.

Texture Analysis. In this step, the Gray Level Co-occurrence Matrix (GLCM) based on texture feature from each of the selected slices was computed.

The GLCM [8] (Gray Level Co-occurrence Matrix) is a statistical calculation of how often various combination of gray level pixel values occur in an image. GLCM is a matrix that briefly explains the frequency of one gray level appearing in a specified spatial linear relationship with another gray level within the region of investigation.

Graycomatrix calculates the GLCM from a scaled version of the image [21]. Figure 4 shows how graycomatrix calculates various values in the GLCM of the 4-by-5 intensity image. Element (1, 1) in the GLCM holds the value 1 because there is only one example in the image where two, horizontally adjacent pixels have the values 1 and 1. Element (1, 2) in the GLCM holds the value 2 because

there are two examples in the image where two, horizontally adjacent pixels have the values 1 and 2. Graycomatrix continues this processing to fill in all the values in the GLCM.

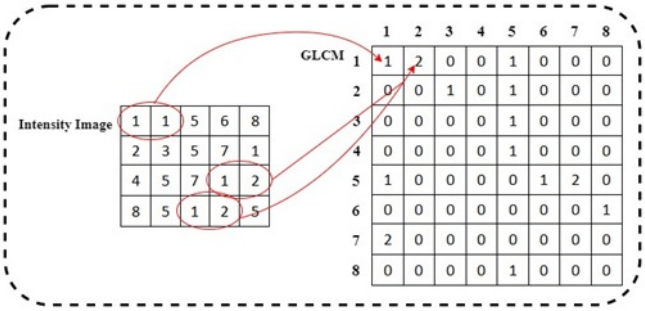


Fig. 4. Graycomatrix calculates several values in the GLCM of the 4-by-5 intensity image

In the approach the GLCM is calculated with Contrast, Correlation, Energy, Homogeneity, Entropy and Mean statistical measures in all four directions considering both type of pairs like $P[i, j]$ and $P[j, i]$. The definitions of these statistical measures are given below.

Contrast. Contrast returns a value after measuring the intensity contrast between a pixel and its neighbor over the entire image. Range of the contrast = [0 (size (GLCM, 1)-1)ˆ2]. Contrast value is always 0 for a constant image. The property of contrast is also known as variance and inertia.

$$Contrast = \sum_{i,j} |i - j|^2 p(i, j)$$

Correlation. Correlation returns a value after measuring of how correlated a pixel is to its neighbor over the entire image. Range of the correlation = [-1 1]. Correlation is 1 or -1 for a perfectly positively or negatively correlated image. Correlation is always NaN for a constant image.

$$Correlation = \sum_{i,j} \frac{(i - \mu i)(j - \mu j)p(i, j)}{\sigma_i \sigma_j}$$

Energy. Energy returns a sum value of squared elements in the GLCM. Range of the energy = [0 1]. Energy is always 1 for a constant image. The property of energy is also known as uniformity, uniformity of energy, and angular second moment.

$$Energy = \sum_{i,j} p(i, j)^2$$

Homogeneity. Homogeneity returns a value that measures the closeness of the distribution of elements in the GLCM to the GLCM diagonal. Range of the homogeneity = [0 1]. Homogeneity is always 1 for a diagonal GLCM.

$$Homogeneity = \sum_{i,j} \frac{p(i,j)}{1+|i-j|}$$

Entropy. Entropy returns a scalar value representing the entropy of grayscale image. Entropy is a statistical measure of randomness that can be used to characterize the texture of the input image.

$$Entropy = \sum_{i,j} -ln(P_{ij})P_{ij}$$

Slice-wise Averaging Attributes. Averaging of the attributes for all the slices has been done to generate the final feature vector. We get 96 attributes feature value from the attributes contrast, correlation, energy, homogeneity, Entropy and Mean.

3.3 Classifiers

In this research work, eight different machine learning classifiers were used to train and classify. They are Bayesian Network (BN), Linear Discriminant Analysis (LDA), Logistic regressiom (L), Fuzzy Unordered Rule Induction Algorithm (FURIA), Random Forest (RF), Random Tree (RT) and J48. A simple voting scheme (Vote) was also experimented. We compare the performance of these classifiers to find the best result. These are briefly described below.

Bayesian Network (BN). For Bayesian Network (BN) [2], we used K2, a hill-climbing technique which is a well known score-based algorithm that recovers the underlying distribution in the form of directed acyclic graph efficiently.

Linear Discriminant Analysis (LDA). Linear Discriminant Analysis (LDA) [1] method easily handles the case where the within-class frequencies are unequal and their performances have been examined on randomly generated test dataset. This technique maximizes the ratio of between the class variance to the within-class variance in any particular dataset thereby guaranteeing maximal separability.

Random Tree (RT). Random tree [18] is a classification algorithm based on supervised classification. It uses the concept of bagging to build a decision tree for constructing a random set of data. This algorithm can handle both classification and regression types. Random trees are a collection of tree predictors. They are amalgamation of two important algorithms used extensively in machine learning namely, single model decision trees and random forest concept.

Logistic (L). Logistic regression is a classification algorithm used to assign observations to a discrete set of classes. Unlike linear regression which outputs continuous number values, logistic regression transforms its output using the logistic sigmoid function to return a probability value which can then be mapped to two or more discrete classes [10].

Fuzzy Unordered Rule Induction Algorithm (FURIA). It is a novel fuzzy rule-based classification method. FURIA extends the famous RIPPER algorithm while preserving its advantages such as simple and comprehensivle rule sets. Also it includes a number of modifications and extensions. Details can be found here [11].

Random Forest (RF). Random forest(RF) [15] is an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

J48. J48 [13] classifier is an extension version of ID3. The additional features of J48 are accounting for missing values, decision trees pruning, continuous attribute value ranges, derivation of rules, etc.

4 Experiments

This chapter includes the dataset description, evaluation metrics and system configuration. Results and analysis are presented at the end of this chapter.

4.1 Dataset Description

This research work was divided into two tasks, both based on lung CT images of patients with tuberculosis. The first task consisted of predicting multi-drug resistant (MDR) patients versus drug-sensitive (DS) cases. The dataset has two classes (MDR and DS) with 230 patients. Table 1 contains the exact number of subjects of multi-drug resistant dataset.

Class	#Patients
DS	134
MDR	96

Table 1. Number of patients per class in the multi-drug resistance dataset

The second task consisted of a multi-class classification problem. It contains 500 patients data with five tuberculosis types: infiltrative, focal, tuberculoma,

miliary, and fibro-cavernous. No information about the relation between this the classes is suggested. The number details of tuberculosis type dataset are specified in Table 2.

Class	#Patients
Infiltrative	140
Focal	120
Tuberculoma	100
Miliary	80
Fibro-cavernous	60

Table 2. Number of patients per class in the tuberculosis type dataset

Moreover, lung segmentation extracted automatically [5] were also provided. In our work, we used these segmentations to restrict the region of interest of the lungs. Figure 5 shows the sample slices of the Computerized Tomography (CT) images with segmented lungs.

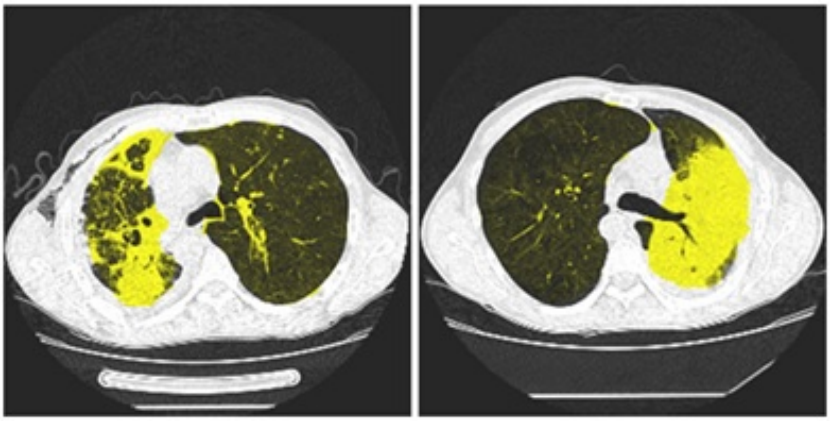


Fig. 5. Sample slices of CT images with segmented lungs [14]

4.2 Evaluation Metrics and System Configuration

For task 1 the performance of the system was measured using the area under the curve of the Receiver Operator Characteristic (AUC) and Accuracy; the ROC curve is created by plotting the true positive rate against the false positive rate. For task 2, we use Kappa Statistic and Accuracy.

To evaluate the system, we used stratified five-fold cross-validation. Regarding the resources, all experiments were carried out using MATLAB 2017b software in a system with 3.5 GHz CPU, 8 GB RAM and Weka 3.8.1 [9].

The rationale for using these measures was its use on the CLEF 2017 corresponding tasks.

4.3 Result and Discussion

In this section, we give a short interpretation of the preliminary evaluation results. We compare the result based on the Area Under the ROC Curve (AUC) for the MDR task and Accuracy. On the other hand, unweight Cohen's Kappa coefficient (Kappa) for the TBT task.

Table 3 presents the results for the multi-drug resistance task, for algorithms mentioned above and for the voting of the 3 best ones. The results are presented in a decreasing order of Accuracy.

AUC scores range from 0.04 to 0.66 and accuracy is in a range from 53.48% to 62.87%. For multi-drug resistance task, the ensemble classifier provides the best results (for Accuracy and ROC Area).

Algorithm	Accuracy	ROC Area (AUC)
Random Tree (RT)	62.61	0.62
Random Forest (RF)	60.87	0.61
Baysian network (BN)	58.27	0.50
J48	56.09	0.50
Linear Discriminant Analysis (LDA)	55.65	0.59
Fuzzy Unordered Rule Induction Algorithm (FURIA)	55.65	0.51
Logistic (L)	53.48	0.04
Vote	62.87	0.66

Table 3. Result of Task 1 – Multi-drug resistance detection

Table 4 presents the list of results for tuberculosis type task. Once again, the results obtained are presented on a decreasing order of Accuracy and the voting scheme is included.

Kappa Statistics value range from 0.059 to 0.222 and accuracy range from 26.2% to 39.0%. In this tuberculosis task, the ensemble classifier also gives the best result (for Accuracy and Kappa Statistics).

In our experiments, ensemble classifier gives us better performance in both tasks.

5 Conclusions and Future Work

In this research, we present a new model using Gray-Level Co-occurrence Matrix (GLCM) for representing medical images. Its discriminating power was then tested using different machine learning classifiers. For the MDR task, we got 62.87% for Accuracy and 0.66 for AUC; for the tuberculosis type, the best result obtained was 39.0% for Accuracy and a value of 0.222 for kappa statistics. In

Algorithm	Accuracy	Kappa Statistics
Logistic (L)	38.6	0.217
Linear Discriminant Analysis (LDA)	36.4	0.177
Random Forest (RF)	32.4	0.131
Baysian network (BN)	30.6	0.129
J48	30.2	0.108
Fuzzy Unordered Rule Induction Algorithm (FURIA)	28.8	0.054
Random Tree (RT)	26.2	0.059
Vote(LDA, L)	39.0	0.222

Table 4. Result of Task 2 – Tuberculosis type classification

both tasks, the best results were obtained using a simple voting scheme. These results were obtained using no patient clinical information, but only features extracted from the 3D CT scans.

In future, we will use patient clinical information to improve the accuracy of both tasks. The best ML algorithms will be tuned and more sophisticated ensemble methods will be tested aiming to increase the performance of the system.

6 Acknowledgment

The authors thank the LEADER (Links in Europe and Asia for engineering, eDucation, Enterprise and Research) Erasmus Mundus project for the scholarship that enabled this work.

References

[1] S. Balakrishnama and A. Ganapathiraju. Linear discriminant analysis-a brief tutorial. *Institute for Signal and information Processing*, 18:1–8, 1998.

[2] C. Bielza, G. Li, and P. Larranaga. Multi-dimensional classification with bayesian networks. *International Journal of Approximate Reasoning*, 52(6):705–727, 2011.

[3] D. Braun, M. Singhof, M. Tatusch, and S. Conrad. Convolutional neural networks for multidrug-resistant and drug-sensitive tuberculosis dinstinction. In *CLEF2017 Working Notes. CEUR Workshop Proceedings, Dublin, Ireland, CEUR-WS. org¡ http://ceur-ws. org¿(September 11-14 2017)*, 2017.

[4] Y. D. Cid, K. Batmanghelich, and H. Müller. Textured graph-model of the lungs for tuberculosis type classification and drug resistance prediction: Participation in imageclef 2017. *In CLEF2017 Working Notes. CEUR Workshop Proceedings, Dublin, Ireland, CEUR-WS. org¡ http://ceur-ws. org¿(September 11-14 2017)*, 2017.

[5] Y. D. Cid, O. A. J. del Toro, A. Depeursinge, and H. Müller. Efficient and fully automatic segmentation of the lungs in ct volumes. In *VISCERAL Challenge@ ISBI*, pages 31–35, 2015.

[6] Y. D. Cid, H. Müller, and K. Batmanghelich. Batmanlab in the imageclef tuberculosis task 2017. *In CLEF2017 Working Notes. CEUR Workshop Proceedings, Dublin, Ireland, CEUR-WS. org¡ http://ceur-ws. org¿(September 11-14 2017)*, 2017.

[7] Y. Dicente Cid, A. Kalinovsky, V. Liauchuk, V. Kovalev, , and H. Müller. Overview of ImageCLEFtuberculosis 2017 - predicting tuberculosis type and drug resistances. In *CLEF2017 Working Notes*, CEUR Workshop Proceedings, Dublin, Ireland, September 11-14 2017. CEUR-WS.org <http://ceur-ws.org>.

[8] C. Halder, S. M. Obaidullah, J. Paul, and K. Roy. Writer verification on bangla handwritten characters. In *Advanced Computing and Systems for Security*, pages 53–68. Springer, 2016.

[9] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: an update. *SIGKDD Explorations*, 11(1):10–18, 2009.

[10] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

[11] J. Hühn and E. Hüllermeier. Furia: an algorithm for unordered fuzzy rule induction. *Data Mining and Knowledge Discovery*, 19(3):293–319, 2009.

[12] J. Kalpathy-Cramer, A. G. S. de Herrera, D. Demner-Fushman, S. Antani, S. Bedrick, and H. Müller. Evaluating performance of biomedical image retrieval systems—an overview of the medical image retrieval task at imageclef 2004–2013. *Computerized Medical Imaging and Graphics*, 39:55–61, 2015.

[13] G. Kaur and A. Chhabra. Improved j48 classification algorithm for the prediction of diabetes. *International Journal of Computer Applications*, 98(22), 2014.

[14] V. Liauchuk and V. Kovalev. Imageclef 2017: Supervoxels and co-occurrence for tuberculosis ct image classification. In *CLEF2017 Working Notes. CEUR Workshop Proceedings, Dublin, Ireland, CEUR-WS. org¡ http://ceur-ws. org¿(September 11-14 2017)*, 2017.

[15] A. Liaw, M. Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[16] H. Müller, P. Clough, T. Deselaers, B. Caputo, and I. CLEF. Experimental evaluation in visual information retrieval. *The Information Retrieval Series*, 32:1–554, 2010.

[17] W. H. Organization et al. Global tuberculosis report 2016. *World Health Organization*, 2016.

[18] D. K. Prasad, L. Vibha, and K. Venugopal. Severity analysis of macular edema using random tree classifier. *International Journal Of Engineering And Computer Science*, 7(03):23674–23679, 2018.

[19] J. F. Silva, J. M. Silva, E. Pinho, and C. Costa. 3d-cnn in drug resistance detection and tuberculosis classification. In *CLEF2017 Working Notes. CEUR Workshop Proceedings, Dublin, Ireland, CEUR-WS. org¡ http://ceur-ws. org¿(September 11-14 2017)*, 2017.

[20] L.-D. Stefan, Y. D. Cid, O. Jimenez-del Toro, B. Ionescu, and H. Müller. Finding and classifying tuberculosis types for a targeted treatment: Medgift–upb participation in the imageclef 2017 tuberculosis task. *In CLEF2017 Working Notes. CEUR Workshop Proceedings, Dublin, Ireland, CEUR-WS. org¡ http://ceur-ws. org¿(September 11-14 2017)*, 2017.

[21] M. Subramanian and S. Sathappan. An efficient content based image retrieval using advanced filter approaches. *International Arab Journal of Information Technology (IAJIT)*, 12(3), 2015.

[22] J. Sun, P. Chong, Y. X. M. Tan, and A. Binder. Imageclef 2017: Imageclef tuberculosis task-the sgeast submission. In *CLEF2017 Working Notes. CEUR Workshop Proceedings, Dublin, Ireland, CEUR-WS. org¡ http://ceur-ws. org¿(September 11-14 2017)*, 2017.

PREDICTING STUDENT PERFORMANCE USING COURSE MANAGEMENT SYSTEMS DATA

Carlos Rodrigues and Irene Rodrigues and Luís Sebastião

Universidade de Évora

Keywords: Educational Data Mining, E-learning, Adaptive Learning, Personalized Learning, User modeling, Intelligent Tutoring System, Learning Management System, Course Management System

Abstract. Over the last few years several institutions have increasingly used course management systems as a means of complementing or extending their courses to new students. The large amounts of data stored through these systems allow the use of machine learning techniques in predicting student performance. This process will be an important help in finding strategies to improve school outcomes. This article presents and compares different classification techniques applied to specific educational datasets and analyzes the results obtained.

1 Introduction

The increasing use of virtual learning environments (VLE) has allowed the storage of large amounts of data with information about students' online behavior. Some of these data have been used by researchers to predict student performance. However, research has led to a wide range of findings about the performance-related mechanisms of students, possibly due to the wide diversity of courses and variables that have been extracted, making it difficult to draw general conclusions.

In this work six machine learning (ML) models were tested(i.e. Decision Trees(DT), Random Forest(RF), Support Vector Machines (SVM), Naive Bayes(NB) and Neural Networks(NN) and two input selections (e.g. with and without previous grades) were tested. The results show that a good predictive accuracy can be achieved, based on VLEs activities.

The main contributions of this article include the comparison of different classification techniques applied to educational datasets for predicting student performance. They are further tested if activity levels, page views or submissions correlate with the results obtained. We use two data sets: MITx-Harvardx dataset [15] and Moodle with administrative data from the University of Évora.

In section 2 related work is presented. Section 3 presents the methods and tools used. Section 4 presents experimental results. Section 5 presents the conclusions and future work.

2 Related Work

Predicting students' performance has already been studied previously in educational data mining research in the context of predicting student attrition or student dropout [7, 14]. In [13] the authors show how web usage mining can be applied in e-learning systems in order to predict the marks that university students will obtain in the final exam of a course. Agnihotri et al [16] applied data mining in this area and showed that there is positive correlation between grades and login activity, but only up to certain level of activity, beyond which the effect diminishes. Lindrum et al. [17] proposed early at-risk factor detection by measuring how well a final grade could be predicted by whether the student opened a course resource within a given time period.

In [4], student grades were predicted by applying various Artificial Neural Networks to Moodle data for 250 students. Features used included the number of examination sessions, mark, total accesses, percentage of resource views, total number of resources of each type viewed, and percentage of accesses per month.

Minaei-Bidgoli [3] used a combination of multiple classifiers to predict their final grade based on features extracted from logged data in an education web based system.

In [10] recommender system techniques are used for educational data mining. They compare recommender system techniques with traditional regression methods such as logistic/linear regression by using educational data for intelligent tutoring systems and concluded that the proposed approach can improve prediction results

In [12] used real data about 670 high school students, and proposed a genetic programming algorithm and different data mining approaches for predicting student failure in order to obtain both more comprehensible and better accuracy classification rules.

Romero et al. [5] focused on comparing different data mining methods and techniques for classifying students based on their Moodle (e-learning system) usage data and the final marks obtained in their respective programs. The conclusion was that the most appropriate algorithm was decision trees for being accurate and comprehensible for instructors.

Kabakchieva [11] developed models for predicting student performance, based on their personal, pre-university and university performance characteristics. The highest accuracy is achieved with the neural network model, followed by the decision tree model and the kNN model.

In [18] supervised learning techniques are applied to a data repository from UE in order to show how it is possible to make predictions about the success of students based on their usage of Moodle.

In general research has shown that certain activity patterns in Learning management Systems (LMS) are an indicator of good student performance [5]. Typically higher levels of activity correlates with good grades [8].

However, research has led to a wide range of findings about the performance-related mechanisms of students, possibly due to the wide diversity of courses and variables that have been extracted, making it difficult to draw general conclusions and much work is to be done in this area.

3 Methods and Tools

There are many applications or tasks in educational environments that have been approached through DM methods and tools. In this section, methods and tools are described.

The experiments were carried out using two different datasets: MITx-Harvardx dataset [15] and Moodle with administrative data from the University of Évora.

The experiments were conducted using Orange, an open-source data visualization, machine learning, and data mining toolkit with Python and Scikit-learn library(machine learning library for the Python programming language).

The algorithms used are k-Nearest Neighbors (kNN) [9], Decision Tree (DT), Random Forest (RF) [2], Support Vector Machines (SVM) [21], Naive Bayes (NB) [5] and Neural Networks (NN). This selection of algorithms was based on the most used algorithms for general data mining problems [6].

KNN is a technique that classifies each record in a dataset based on a combination of the classes of the k record(s) most similar to it in a historical dataset (where k is greater than or equal to 1). DT is tree shaped structures that represent sets of decisions. These decisions generate rules for the classification of a dataset. Random forest is an ensemble learning method used for classification, regression and other tasks [2]. Support vector machine (SVM) is a machine learning technique that separates the attribute space with a hyper plane, thus maximizing the margin between the instances of different classes or class values. NB classifiers are probabilistic models based on Bayes theorem witch describes the probability of an event, based on prior knowledge of conditions that might be related to the event. Neural network is a set of connected input/output units and each connection has a weight present with it. During the learning phase, network learns by adjusting weights so as to be able to predict the correct class labels of the input tuples. Neural networks have the remarkable ability to derive meaning from complicated or imprecise data and can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques.

In this set of experiments default values of parameters was used. Models were evaluated using 10 fold cross-validation method with stratified sampling [1].

The performance analysis of the various classification techniques was carried out through a confusion matrix. The base structure of a confusion matrix is presented in Table 1.

	Classified Positive	Classified Negative
Actual Positive	TP	FN
Actual Negative	FP	TN

Table 1. Confusion Matrix

The columns represent the classifier prediction and the rows are the actual classes. In the confusion matrix, TP (True Positive) is the number of positive cases correctly classified as such. FN (False Negative) is the number of positive cases incorrectly classified as negatives. FP (False Positive) is the number of negative cases that are incorrectly identified as positive cases and TN (True Negative) is the number of negative cases correctly classified as such. Classification accuracy (CA) is the proportion of correctly classified examples (equation 1).

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

4 Experimental Results

Experiments were carried out in order to evaluate the performance of the different algorithms for predicting student performance.
Before testing the models, some preprocessing tasks were required such as cleaning, integration, discretization and variable transformation.

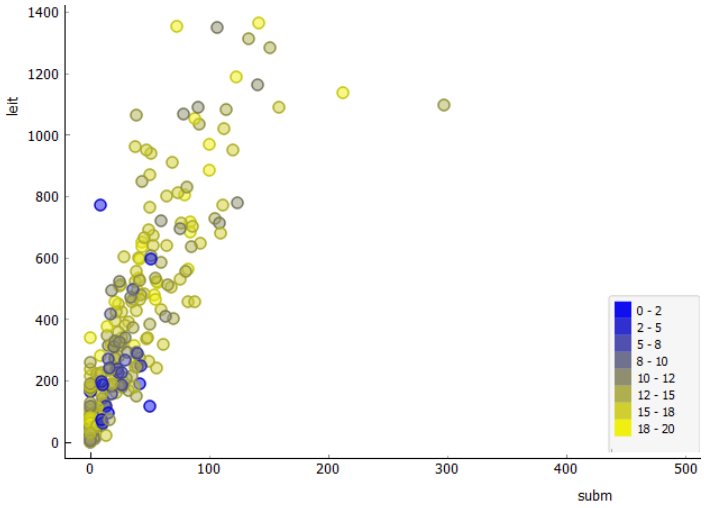


Fig. 1. Correlation nº readings/ nº submissions in UE Dataset

During this process those students without complete information were eliminated and calculations were made from the log data. The resulting dataset Moodle-Évora included 249 instances with online features (e.g. number of readings, number of submissions, number of weeks, final grade). The resulting dataset MITx included 525296 instances with online features (e.g. number days active, number of video views, number of forum posts, number chapters viewed and final grade). Student grades were modeled using three classification approaches (i.e. binary(A); 3-level (B) and 4-level (C)).

First we have construct some scatterplots. A scatterplot is a graph that is used to plot the data points for two variables.
In Figure 1 we observe that number of readings is positively correlated with the number of submissions.
After the pre-processing step, the tests were performed with the selected algorithms. In all tests the division between the training set and the testing set was done through Cross Validation with the number of 10 folds.
Tables 2 and 3 show the average performance obtained with different classification algorithms, using binary classification (A); 3-level classification (B) and 4-level classification (C) in two datasets:

	Models					
Classification Approaches	RF	KNN	DT	SVM	NB	NN
A	0.863	0.88	0.823	**0.874**	0.84	0.87
B	0.699	0.699	0.703	**0.723**	0.635	0.707
C	**0.602**	0.594	0.546	0.562	0.482	0.586

Table 2. Accuracy of UE Dataset

	Models					
Classification Approaches	RF	KNN	DT	SVM	NB	NN
A	**0.986**	0.983	0.972	0.971	0.943	0.98
B	0.978	0.973	0.963	0.945	0.927	**0.979**
C	0.968	0.966	0.958	0.941	0.924	**0.971**

Table 3. Accuracy of MITx Dataset

Results show that there is not one algorithm that obtains significantly better classification accuracy. MITx Dataset, with a bigger sample size, presented better results for the tree approaches because of more imbalanced data in UE dataset with few samples in some classes.

5 Discussion and Conclusion

Although there are so many benchmarks comparing the performance and accuracy of different classification algorithms, there are still very few experiments carried out on Educational datasets. In this work, we compare the performance of six data mining algorithms in two different datasets: UE Dataset and MITx Dataset
The selected algorithms: k-Nearest Neighbors, Decision Tree, Random Forest, Support Vector Machines, Naive Bayes (NB) and Neural Networks have shown that classification algorithms can be used successfully

in order to predict a student's academic performance in particular, to model the difference between Fail and Pass students. Besides, our experimentation shows that there is not one algorithm that obtains significantly better classification accuracy. In fact, the accuracy depends on the sample size and the number of level classification, and the type of attributes.
Our near future work is to extend this experimentation with different input setups and ensemble algorithms to validate these conclusions and next, to apply recommender methods in predicting the students' success.

References

[1] Michael W. Browne. "Cross-validation methods". In: *Journal of Mathematical Psychology* 44.1 (2000), pp. 108–132.

[2] Leo Breiman. "Random forests". In: *Machine Learning* 45.1 (2001), pp. 5–32.

[3] Behrouz Minaei-Bidgoli et al. "Predicting student performance: an application of data mining methods with an educational web-based system". In: *Frontiers in education, 2003. FIE 2003 33rd annual*. Vol. 1. IEEE. 2003.

[4] M. Delgado Calvo-Flores et al. "Predicting students' marks from Moodle logs using neural network models". In: *Proceedings of the IV international conference on multimedia and information and communication technologies in education (MICTEE2006)* 1 (2006), pp. 586–590.

[5] Cristóbal Romero, Sebastián Ventura, and Enrique García. "Data mining in course management systems: Moodle case study and tutorial". In: *Computers and Education* 51.1 (2008), pp. 368–384.

[6] Xindong Wu et al. "Top 10 algorithms in data mining". In: *Knowledge and Information Systems* 14.1 (2008), pp. 1–37.

[7] Amelia Zafra and Sebastian Ventura. "Predicting Student Grades in Learning Management Systems with Multiple Instance Genetic Programming." In: *International Working Group on Educational Data Mining* (2009).

[8] Kevin Casey and J Paul Gibson. "(m)Oodles of Data: Mining moodle to understand student behaviour". In: *Proceedings of the 10th International Conference on Engaging Pedagogy (ICEP10)*. Vol. 2010. m. 2010, pp. 61–71.

[9] Tuomas Tanner, Hannu Toivonen, and Typing Master. "Predicting and preventing student failure using the k -nearest neighbour method to predict student performance in an online course environment". In: *International Journal of Learning Technology* 5 (2010), pp. 356–377.

[10] Nguyen Thai-Nghe et al. “Recommender system for predicting student performance”. In: *Procedia Computer Science*. Vol. 1. 2. 2010, pp. 2811–2819.

[11] Dorina Kabakchieva. “Predicting student performance by using data mining methods for classification”. In: *Cybernetics and Information Technologies* 13.1 (2013), pp. 61–72.

[12] Carlos Márquez-Vera et al. “Predicting student failure at school using genetic programming and different data mining approaches with high dimensional and imbalanced data”. In: *Applied Intelligence* 38.3 (Aug. 2013), pp. 315–330.

[13] Cristobal Romero et al. “Web usage mining for predicting final marks of students that use Moodle courses”. In: *Computer Applications in Engineering Education* 21.1 (2013), pp. 135–146.

[14] Lalitha Agnihotri and Alexander Ott. “Building a Student At-Risk Model : An End-to-End Perspective”. In: *Proceedings of the 7th International Conference on Educational Data Mining (EDM)* (2014), pp. 209–212.

[15] MITx and HarvardX. “HarvardX-MITx Person-Course Academic Year 2013 De-Identified dataset, version 2.0”. In: (2014).

[16] Lalitha Agnihotri et al. “Mining Login Data For Actionable Student Insight”. In: *Proceedings of the 8th International Conference on Educational Data Mining (EDM)*. 2015, pp. 472–475.

[17] RS Baker and D Lindrum. “Analyzing Early At-Risk Factors in Higher Education e-Learning Courses”. In: *Students at Risk: Detection and Remediation* (2015).

[18] Pedro Miguel Lúcio Melgueira. “Educational data mining applied to Moodle data from the University of Évora”. MA thesis. Universidade de Évora, 2017.

Teste Automático de Softwares Públicos

Vanderley Gondim[1]

[1]Universidade de Evora, Évora, Portugal
d11194@alunos.uevora.pt

Resumo. Com o passar dos anos, o *software* vem representando um papel fundamental na rotina da Administração Pública e em especial na prestação de serviços ao cidadão. A busca por alternativas funcionais e de baixo custo em órgãos da máquina estatal que possam ser utilizadas pela administração pública em geral, tornou-se imperativo. Com os objetivos de redução de gastos, minimizar a multiplicidade e a redundância de esforços, racionalizar a gestão dos recursos de informática, melhorar o atendimento à população e criar espaços de colaboração com a sociedade, o Software Público Brasileiro (SPB) surge nesse cenário como aliado. Para aprimorar a qualidade dos produtos de software disponíveis no Portal do *Software* Público Brasileiro (PSPB) algumas ferramentas de testes são utilizadas, mas há uma em particular que não faz parte desse processo, o gerador de testes automáticos *Randoop*. Como os *softwares* do PSPB não passam por testes automáticos, perde-se a oportunidade de encontrar defeitos antes de sua publicação e ainda serem executados de forma rápida e sem custos na sua aquisição. O objetivo deste artigo é analisar a qualidade dos *softwares* disponíveis no PSPB e propor a utilização da ferramenta livre *Randoop* na realização dos testes.

Palavras-chave: Teste de Software, Software Livre, Software Público, Randoop.

1 Introdução

O software vem representando um papel fundamental também na rotina da Administração Pública e em especial na prestação de serviços ao cidadão. Com os objetivos de redução de gastos, minimizar a multiplicidade e a redundãncia de esforços, racionalizar a gestão dos recursos de informática, melhorar o atendimento à população e criar espaços de colaboração com a sociedade, o Software Público Brasileiro (SPB) surge nesse cenário como um forte aliado. O SPB foi criado em 2007 com o intuito inicialmente de incentivar o uso e desenvolvimento de software livre pelos governos e torná-lo disponível como um bem público. Em pouco tempo, uma extensa comunidade formou-se em torno da solução, o que serviu de base para a definição do conceito de Software Público e para a sua materialização através do Portal do Software Público Brasileiro (PSPB)[1]. Mais do que apenas um ambiente de desenvolvimento de software compartilhado, o Portal do Software Público Brasileiro

[1] http://www.softwarepublico.gov.br

congrega diversos atores, desde usuários e desenvolvedores a prestadores de serviço sob a mesma plataforma compartilhada. De acordo com Freitas [1], para a rede de atores que constitui o Portal SPB, é importante o crescente acúmulo de capital tecnológico-informacional, definido como o conjunto de disposições – materiais e imateriais – necessárias para a inserção do indivíduo na sociedade do conhecimento.

Cada software, para ser classificado como bem público e estar disponível no Portal, precisa estar associado a uma série de serviços, como guia do usuário, manual de instalação, sites de discussão on-line, fóruns, diretrizes para testes e qualidade, governança e apoio [2]. Todos os procedimentos para o desenvolvimento, a disponibilização e o uso do SPB foram disciplinados pela Instrução Normativa nº 01, de 17/01/2011, posteriormente substituída pela portaria nº 46/2016 [3].

Mais de 70 soluções para as áreas de Educação, Gestão de Tecnologia da Informação e Gestão Pública já foram disponibilizadas no Portal do Software Público, com mais de 170 mil usuários e 200 empresas em todo o Brasil cadastradas como prestadores de serviços para essas soluções [4].

2 Metodologia

Para o desenvolvimento deste trabalho, promoveu-se uma revisão da literatura nos principais indexadores, identificando, selecionando e avaliando as atuais publicações relacionados com o tema proposto. À partir do acesso ao Portal do *Software* Público Brasileiro (PSPB), foram identificados 5 softwares para a realização de testes estruturais utilizando a ferramenta livre Randoop. A ferramenta livre Randoop testa apenas softwares baseados em Java. Todas as classes de cada *software* foram testadas e os erros coletados, identificados e exibidos através de gráficos.

3 Objetivo

O objetivo deste trabalho é analisar a qualidade de softwares disponibilizados ao público pelo Portal do Software Público do Governo Federal Brasileiro e propor a utilização da ferramenta livre Randoop na realização de testes estruturais (caixa-branca).

4 Teste de Software

Durante o processo de desenvolvimento de um *software* existem atividades que procuram garantir a qualidade do produto final; entretanto, apesar dos métodos, técnicas e ferramentas utilizadas, falhas no produto ainda podem ocorrer. Assim, a etapa de teste, a qual representa uma das atividades de garantia de qualidade, é de grande importância para a identificação e eliminação de falhas, representando assim o último passo do desenvolvimento do *software* [5].

De acordo com Myers [6], Teste de *Software* é o processo de execução de um programa com o intuito de encontrar erros. Escolher uma técnica de teste, depende a

princípio de uma série de fatores, como o tipo de sistema, padrões, clientes, requisitos contratuais, nível do risco, tipos de riscos, objetivos do teste, documentação disponível, conhecimento dos testadores, tempo, dinheiro, ciclo de desenvolvimento, modelo de caso de uso e uma experiência prévia do tipo de defeitos encontrados [7].

Algumas técnicas são mais facilmente aplicadas em certas situações e níveis de teste, já outras são aplicáveis a todos os níveis.

Segundo o ISTQB [8], podemos classificar os testes de softwares da seguinte forma:

Técnicas de caixa preta (também chamadas de técnicas baseadas em especificação), são uma forma de derivar e selecionar as condições e casos de testes baseados na análise da documentação, seja funcional ou não-funcional, para um componente ou sistema sem levar em consideração a sua estrutura interna. Não se faz uso do código-fonte para criar testes, seja porque o código não está disponível, seja porque não se quer ser influenciado pela implementação.

De acordo com Fournier [9], teste de caixa preta refere-se a testar um item de software sem saber nada sobre seu funcionamento interno - sobre como ele faz o trabalho! O sistema sob teste é realmente tratado como uma caixa preta. O teste caixa-branca objetiva verificar se a estrutura interna da unidade está correta. Esta verificação é efetuada através de casos de teste que visam percorrer todos os caminhos internos possíveis da unidade [10].

5 Ferramenta Livre Randoop

O Randoop (Testador Randômico para Programas Orientados a Objeto) [11], é um gerador de teste de unidade automático para ambiente Java. Ele cria automaticamente testes de unidade para suas classes em formato JUnit [12]. Randoop gera testes de unidade usando geração de teste aleatório dirigido por *feedback*. Em poucas palavras, esta técnica testa de forma aleatória e de forma inteligente, gera sequências de métodos e as invocações do construtor para as classes em teste, e usa as sequências para criar testes.

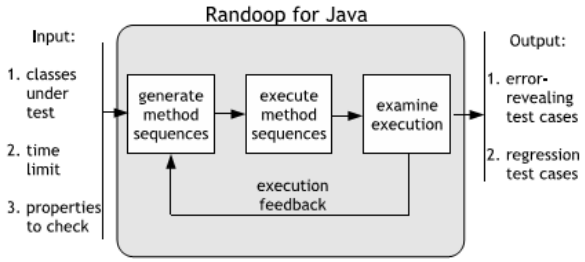


Fig. 1: Funcionamento do Randoop
Fonte: Pacheco e Ernst [13]

Em seguida, o Randoop executa as sequências que cria, utilizando os resultados da execução para criar assertivas que capturam exceções levantadas pelos programas, conforme Fig. 1. O Randoop criou testes que encontraram erros previamente desconhecidos de bibliotecas amplamente utilizados, incluindo IBM e JDKs (*Java Development* KIT) da Oracle.

Pacheco e Ernst [13] reforçam esse conceito, afirmando que o Randoop gera testes unitários usando testes aleatórios direcionados por *feedback*, uma técnica inspirada em testes aleatórios que usa *feedback* de execução coletado da execução de entradas de teste à medida que são criadas, para evitar a geração de entradas redundantes e ilegais. O Randoop cria sequências de métodos de forma incremental, selecionando aleatoriamente uma chamada de método para aplicar e selecionar argumentos de sequências construídas anteriormente.

Assim que ele é criado, uma nova sequência é executada e verificada contra um conjunto de contratos. As sequências que levam a violações de contrato são enviadas para o usuário como testes de contratualização [14].

6 Software Público

O Portal do Software Público Brasileiro, foi criado em 12 de abril de 2007 com o intuito de compartilhar soluções de software a todos os setores da sociedade, sem custos e com suporte das comunidades que se formaram no seu entorno. De acordo com o Portal do Software Público Brasileiro [4], o Software Público Brasileiro é um tipo específico de software que adota um modelo de licença livre para o código-fonte, a proteção da identidade original entre o seu nome, marca, código-fonte, documentação e outros artefatos relacionados por meio do modelo de Licença Pública de Marca – LPM e é disponibilizado na internet em ambiente virtual público denominado Portal do Software Público Brasileiro - PSPB. Meirelles [15] e Alves [16] descrevem o PSPB como uma plataforma integrada de desenvolvimento de software baseada na integração e evolução das ferramentas FLOSS (Free/Libre and Open Source Software) existentes, fornecendo vários recursos modernos para desenvolvimento colaborativo de software, ajudando a administração pública brasileira a compartilhar suas soluções. Os serviços disponíveis são acessados até por outros países, como Uruguai, Argentina, Portugal, Venezuela, Chile e Paraguai.

Fig. 2: Portal do Software Público Brasileiro
Fonte: Portal do Software Público Brasileiro [4]

Para submeter um software a ser disponibilizado no Portal do Software Público, é necessário ler a Portaria nº 46/2016 e também o Manual do Ofertante encontrados no Portal do Software Público. O Manual do Ofertante é um guia que orienta como enviar um *software* para o Portal do *Software* Público. Nele, consta a descrição dos critérios de aceitação, dos impedimentos e também dos artefatos e anexos de ofícios necessários para um software se tornar um Software Público.

7 Teste Automático de Software Público

7.1 Questões de Pesquisa

- Técnicas de teste aleatório encontram defeitos nos softwares do PSP?
Sim. Os testes realizados com os softwares integrantes do PSP mostraram que vários erros foram encontrados.
- Em quantos porcento dos sistemas testados havia defeitos?
Dos sistemas testados, em 100% deles foram encontrados defeitos.
- Dos sistemas que apresentam defeitos, em quanto tempo foi necessário testar?
Os sistemas foram testados 10 vezes em intervalos de 1 a 10 minutos cada.
- Qual o tipo de defeito mais comum encontrado?
O defeito mais comum encontrado nos testes foi *java.lang.NullPointerException*.

7.2 Sujeitos

7.2.1 GEPLANES

O Geplanes é um *software* de gestão estratégica elaborado para empresas públicas ou privadas. Ele é utilizado na fase de elaboração do planejamento estratégico e na execução das ações. O Geplanes possibilita gerenciar as medidas, as metas e seus desdobramentos, os indicadores e as anomalias em projetos [4].

Quantidade de classes: 382 classes.

7.2.2 ASES

O ASES – Avaliador e Simulador de Acessibilidade de Sítios – é uma ferramenta que permite avaliar, simular e corrigir a acessibilidade de páginas, sites e portais. [4].

Quantidade de classes: 9 classes.

7.2.3 Sistema Ouvidoria

O Serviço Federal de Processamento de Dados (Serpro) disponibilizou às ouvidorias de administrações públicas, gratuitamente, o Sistema de Ouvidoria que adota internamente. Tal Sistema, desenvolvido em plataforma web, garante adaptabilidade e viabilidade econômica e técnica às ouvidorias, permitindo a emissão de relatórios gerenciais, apresentando estatísticas dos dados consolidados e possibilitando seu uso por ouvidorias com diferentes estruturas. [4].

Quantidade de classes: 261 classes.

7.2.4 EDITOM

O EdiTom é um software de edição de partituras que permite aos iniciantes criar sons, representá-los de forma gráfica e ouvir efeitos sonoros, procurando sempre ter ações reais como ponto de partida para o mundo técnico da música. [4].

Quantidade de classes: 611 classes.

7.2.5 Sistema de Gestão de Frotas

O SGF (Sistema de Gestão de Frotas) foi desenvolvido com o objetivo de otimizar o controle da frota municipal em todos os órgãos da administração pública através de um ambiente único baseado em software livre. [4].

Quantidade de classes: 221 classes.

7.3 Configuração

Para a realização dos testes em cada *Software* Público, foram executados os seguintes procedimentos:

1) Fazer o download do código fonte de cada software no Portal do Software Público;
2) Executar script para localizar e listar todas as classes;
3) Resolver todas as dependências do projeto (localizar biblioteca de terceiros);
4) Executar o comando para cada tempo (de 1 a 10 minutos):
   java -ea -classpath randoop-all-3.0.7.jar:editom_1.jar randoop.main.Main gentests --classlist=myclasses.txt --timelimit=600 --ignore-flaky-tests de acordo com o programa a ser testado, onde:
   classlist = arquivo .txt com a lista de todas as classes encontradas.
   timelimit= tempo em segundos.
   ignore-flaky-tests (booleano)= Se for falso, o Randoop para e fornece diagnóticos sobre *testes flaky* - testes que se comportam de forma diferente em execuções diferentes. Se for verdadeiro, Randoop ignora e não exibe a saída.
5) Executar os testes unitários gerados pelo randoop e coletar os erros.

7.4 Resultados

Os resultados dos testes realizados com os Softwares Públicos são mostrados a seguir:

7.4.1 GEPLANES

Depois de realizados os testes no GEPLANES, identificou-se que nas 10 execuções os erros encontrados foram: *java.lang.NullPointerException* e *java.lang.AssertionError*.

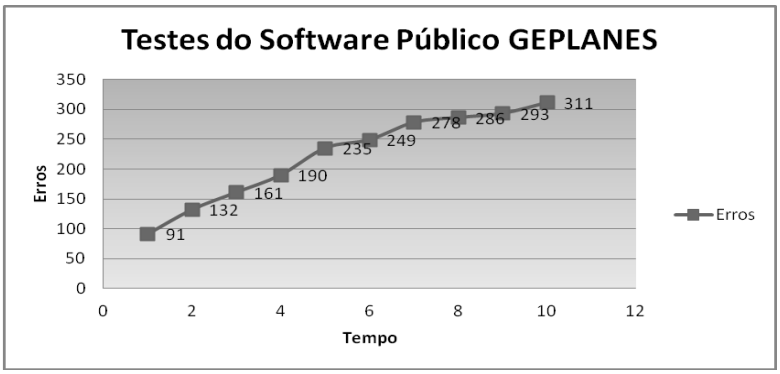


Fig. 4: Testes realizados no GEPLANES
Fonte: Dados do autor

7.4.2 ASES

Depois de realizados os testes no ASES, identificou-se que nas 10 execuções os erros encontrados foram: *java.lang.NullPointerException*.

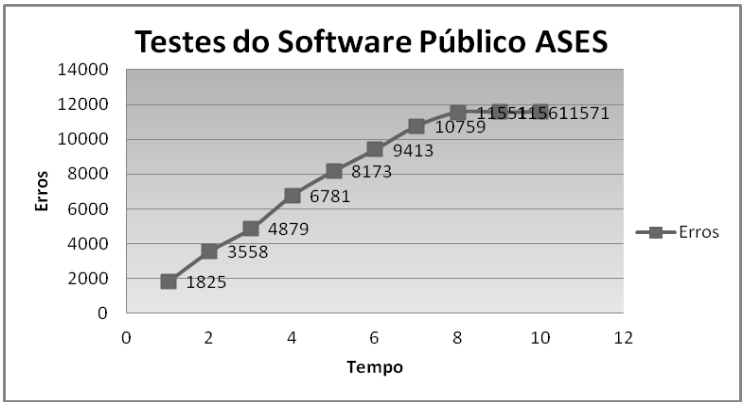


Fig. 5: Testes realizados no ASES
Fonte: Dados do autor

7.4.3 Sistema Ouvidoria

Depois de realizados os testes no Sistema Ouvidoria, identificou-se que nas 10 execuções os erros encontrados foram: *java.lang.ClassCastException*, *java.lang.NullPointerException* e *java.lang.AssertionError*.

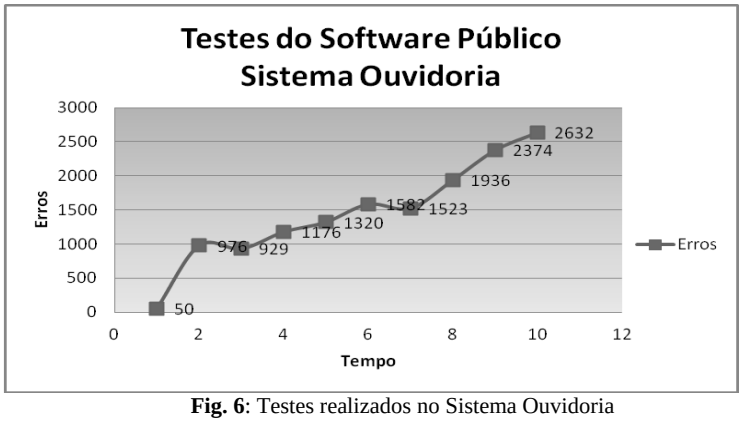


Fig. 6: Testes realizados no Sistema Ouvidoria
Fonte: Dados do autor

7.4.4 EDITOM

Depois de realizados os testes no EDITOM, identificou-se que nas 10 execuções os erros encontrados foram*: java.lang.NullPointerException, java.lang.AssertionError, java.lang.ExceptionInInitializerError, java.lang.NoClassDefFoundError, java.lang.No ClassDefFoundErro, java.lang.UnsupportedOperationException* e *java.lang.stackoverflowerror*.

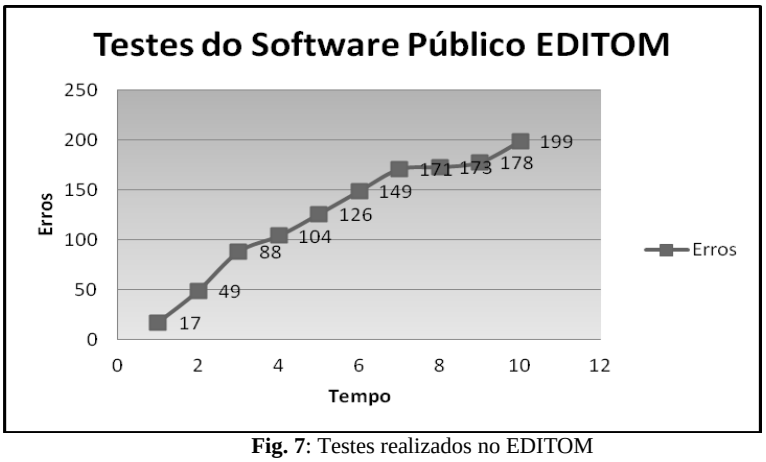


Fig. 7: Testes realizados no EDITOM
Fonte: Dados do autor

7.4.5 Sistema de Gestão de Frotas (SGF)

Depois de realizados os testes no Sistema de Gestão de Frotas, identificou-se que nas 10 execuções os erros encontrados foram*:* *java.lang.NullPointerException* e *java.lang.AssertionError*.

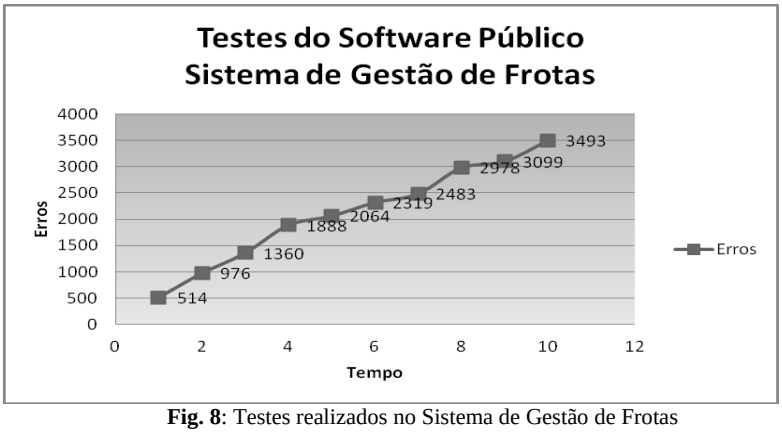


Fig. 8: Testes realizados no Sistema de Gestão de Frotas
Fonte: Dados do autor

8 Discussões e Trabalhos Futuros

Este trabalho de pesquisa foi realizada com o intuito de analisar a qualidade de *softwares* disponibilizados ao público pelo Portal do Software Público do Governo Federal. Para a análise de qualidade, foi utilizado a ferramenta livre de testes aleatórios Randoop. O Randoop foi escolhido para os testes pelo seu grau de efetividade nos testes estruturais (caixa-branca) e por não ter custo em sua aquisição.

No início da pesquisa foram apresentados o contexto, a motivação, o problema de pesquisa, os objetivos geral e específicos que se pretendiam alcançar e as delimitações da pesquisa. Foi realizado uma revisão da literatura, no intuito de apresentar os conceitos principais que fazem parte da pesquisa. Na sequência, foram apresentados os softwares públicos que passaram pelos testes, seu propósito, características e os respectivos resultados dos testes. Em resumo, dos 5 softwares testados, todos apresentaram defeitos. Em média, foram 7 defeitos. O tipo de defeito mais encontrado foi *java.lang.NullPointerException*.

Embora o objetivo da pesquisa tenha sido alcançado, apresentamos abaixo algumas sugestões e recomendações para trabalhos futuros:

- Utilizar outras ferramentas de teste aleatório e analisar o seu grau de efetividade diante dos Softwares Públicos disponíveis no Portal do Software Público;
- Realizar procedimentos de testes nos Softwares Públicos que não foram contemplados nesta pesquisa;
- Como as classes dos softwares relatados na pesquisa foram todas testadas dez vezes em intervalos de tempo de 1 a 10 minutos, sugere-se um estudo sobre os mes-

mos testes levando-se em consideração que cada classe será testada individualmente no mesmo intervalo de tempo.

Referências

1. Freitas, C. S.. O Software Público Brasileiro: novos modelos de cooperação econômica entre Estado e Sociedade Civil. Informação & Sociedade: Estudos, João Pessoa, v. 22, n. 2, p.99-113 (2012).
2. Affonso, L. C. Comunidades de práticas na internet: um estudo de duas comunidades hospedadas em portais públicos brasileiros. 2012. Dissertação – UFRJ, Rio de Janeiro (2012).
3. Brasil. Secretaria de Tecnologia da Informação. Portaria nº 46, de 28 de Setembro de 2016. Disponibilização de Software Público Brasileiro**.** Diário Oficial da República Federativa do Brasil, Poder Executivo, Brasília, DF (2016).
4. Portal do Software Público Brasileiro, https://softwarepublico.gov.br/social/.
5. Inthurn, C. Qualidade e Teste de Software. Visualbooks Editora. Florianópolis-SC. (2001).
6. Myers, G. J. The art of software testing. 3rd ed. John Wiley & Sons, Inc., Hoboken, New Jersey (2012).
7. Copeland, L. A Practitioner's Guide to Software Test Design, Artech House. Boston (2004).
8. International Software Testing Qualifications Board, http://www.istqb.org/downloads/viewcategory/48.html.
9. Fournier, G. Essential Software Testing - A Use-Case Approach. CRC Press. Boca Raton (2009).
10. Henard, C. Papadakis, M. Harman M. Jia Y. Le Traon Y. Comparing White-box and Black-box Test Prioritization. IEEE/ACM 38th IEEE International Conference on Software Engineering *ICSE '16, May 14-22, 2016, Austin, TX, USA.* DOI: http://dx.doi.org/10.1145/2884781.2884791 (2016).
11. Pacheco, C. Directed Random Testing. Ph.D Thesis. Massachusetts Institute of Technology (MIT) (2009).
12. Tahchiev, P. Leme, F.; Massol, V.; Gregory, G. JUnit in Action. 2nd Edition. Manning Publications Co. Greenwich, CT, USA (2010).
13. Pacheco, C. Ernst, M: Randoop: Feedback-Directed Random Testing for Java. In: Object-Oriented Programming Systems, Languages, and Applications - OOPSLA 2007. ACM, Montreal (2007).
14. Pacheco, C.; Lahiri, S. K.; Ball, T.: Finding Errors in .NET with Feedback-Directed Random Testing. In: International Symposium on Software Testing and Analysis - ISSTA'08. ACM, Washington (2008).
15. Meirelles, P.: Brazilian Public Software Portal: an integrated platform for collaborative development. In: 13th International Symposium on Open Collaboration. ACM, Galway (2017).
16. Alves, A. M.: Learning path to an emergent ecosystem: the Brazilian public software experience. In: International Conference on Management of Emergent Digital EcoSystems. ACM, Lyon (2009).