



**Universidade do Minho**  
Escola de Engenharia  
Departamento de Informática

Beatriz de Freitas Rocha

## **Automation of companies' recruitment process**

**Development of an algorithm capable of ranking  
CVs according to job offers**

December 2022



**Universidade do Minho**

Escola de Engenharia

Departamento de Informática

Beatriz de Freitas Rocha

## **Automation of companies' recruitment process**

**Development of an algorithm capable of ranking  
CVs according to job offers**

Master dissertation

Integrated Master's in Informatics Engineering

Dissertation supervised by

**Pedro Rangel Henriques**

**Hugo Paulino Santos**

December 2022

## AUTHOR COPYRIGHTS AND TERMS OF USAGE BY THIRD PARTIES

This is an academic work which can be utilized by third parties given that the rules and good practices internationally accepted, regarding author copyrights and related copyrights.

Therefore, the present work can be utilized according to the terms provided in the license bellow.

If the user needs permission to use the work in conditions not foreseen by the licensing indicated, the user should contact the author, through the RepositóriUM of University of Minho.

**License provided to the users of this work**



**Attribution-NonCommercial**

**CC BY-NC**

<https://creativecommons.org/licenses/by-nc/4.0/>

### STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

Beatriz de Freitas Rocha

---

---

## ABSTRACT

---

This document presents a Thesis and describes the underlying work which was developed along the second year of the Master Degree in Informatics Engineering offered by Departamento de Informática of Universidade do Minho and accomplished at Syone SBS Software – Tecnologia e Serviços de Informática, S.A..

In the past few years, some attempts to automatically screening CVs with resource to Natural Language Processing have been made not only to save recruiters' time, but also to spare them the most tedious task of the recruitment process and, consequently, smooth their job. However, the majority is still very primitive, misclassifies a lot of CVs and needs a deeper study.

Therefore, the aim of this Master's Project is precisely to develop an algorithm that is capable of automatically ranking candidates' CVs according to their similarity regarding the job offer they applied for.

Thus, a general architecture was proposed where CVs and job offers are preprocessed, in order to obtain the respective texts proper to be further processed. That said, two different approaches were followed, in order to find the similarity between the documents in question. To do so, the first approach resorted to several Machine Learning algorithms and similarity measures, while the second approach structured the initial documents to compare their respective information.

After that, tests were conducted to evaluate both approaches and enable the comparison between them. Finally, the conclusions were drawn and also reported in this dissertation.

**KEYWORDS** *Curriculum Vitae* (CV), CV screening, Machine Learning (ML), Natural Language Processing (NLP).

---

## RESUMO

---

Este documento apresenta uma Tese e descreve o trabalho subjacente que foi desenvolvido ao longo do segundo ano do Mestrado em Engenharia Informática do Departamento de Informática da Universidade do Minho e realizado na Syone SBS Software – Tecnologia e Serviços de Informática, S.A..

Nos últimos anos, algumas tentativas de triagem automática de currículos com recurso a Processamento de Linguagem Natural foram feitas não só para economizar o tempo dos recrutadores, mas também para os poupar da tarefa mais entediante do processo de recrutamento e, conseqüentemente, suavizar o seu trabalho. Contudo, a maioria ainda é muito primitiva, classifica incorretamente muitos currículos e necessita de um estudo mais aprofundado.

Sendo assim, o objetivo deste Projeto de Mestrado é precisamente desenvolver um algoritmo capaz de classificar automaticamente os currículos dos candidatos de acordo com a sua similaridade relativamente à oferta de emprego a que se candidataram.

Deste modo, foi proposta uma arquitetura geral onde os CVs e as ofertas de emprego são pré-processados, de forma a obter os respetivos textos adequados para posterior processamento. Dito isto, foram seguidas duas abordagens distintas, de forma a encontrar a semelhança entre os documentos em questão. Para tal, a primeira abordagem recorreu a diversos algoritmos de Aprendizagem Automática e medidas de similaridade, enquanto a segunda abordagem estruturou os documentos iniciais para comparar as suas respetivas informações.

De seguida, foram realizados testes para avaliar ambas as abordagens e possibilitar a comparação entre elas. Por fim, as conclusões foram tiradas e também relatadas nesta dissertação.

**PALAVRAS-CHAVE** Aprendizagem Automática, *Curriculum Vitae* (CV), Processamento de Linguagem Natural (PLN), triagem de currículos.

---

## CONTENTS

---

<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
1.1	Context	1
1.2	Motivation	2
1.3	Objectives	2
1.4	Research hypothesis	2
1.5	Research methodology	3
1.6	Document structure	3
<b>2</b>	<b>STATE OF THE ART</b>	<b>5</b>
2.1	CV sources and formats	5
2.1.1	Europass	5
2.1.2	LinkedIn profile as a CV	8
2.1.3	Curriculum Lattes	11
2.1.4	CIÊNCIAVITAE	14
2.1.5	Syone's internal CV	25
2.1.6	Considerations	28
2.1.7	Ontology definition	28
2.2	Semantic information extraction	31
2.2.1	Entities and relationships	31
2.2.2	Information Extraction workflow	32
2.2.3	Named Entity Recognition approaches	34
2.2.4	Information Extraction workflow with spaCy	34
2.3	Similarity algorithms	39
2.3.1	Jaccard Similarity	39
2.3.2	TF-IDF	40
2.3.3	Doc2Vec	40
2.3.4	BERT	42
2.3.5	USE	44
2.3.6	Algorithm comparison	44
2.4	Similarity measures	45
2.4.1	Cosine similarity	45
2.4.2	Euclidean distance	46
2.4.3	Manhattan distance	46
2.4.4	Chebyshev distance	47
2.5	Summary	47

3	PROPOSED APPROACH	48
3.1	System architecture	48
3.2	Summary	51
4	CVS PROCESSING	52
4.1	Text extractor and XML extractor	52
4.2	Knowledge extractor	53
4.3	Document builder	64
4.4	Summary	64
5	SIMILARITY ALGORITHM (APPROACH A)	65
5.1	Implementation and results	65
5.2	Summary	79
6	SIMILARITY ALGORITHM (APPROACH B)	80
6.1	Implementation and results	80
6.2	Summary	89
7	CONCLUSION	90
A	SAMPLE JOB OFFER	97
B	SAMPLE CV	99
C	JSON FILE CORRESPONDING TO THE SAMPLE JOB OFFER	108
D	JSON FILE CORRESPONDING TO THE SAMPLE CV	112



---

## LIST OF FIGURES

---

Figure 1	Europass - Personal Information	6
Figure 2	Europass - Work Experience	6
Figure 3	Europass - Education and Training	6
Figure 4	Europass - Personal Skills (part 1)	7
Figure 5	Europass - Personal Skills (part 2)	7
Figure 6	Europass - Additional Information	8
Figure 7	LinkedIn profile as a CV - Summary	8
Figure 8	LinkedIn profile as a CV - Experience	8
Figure 9	LinkedIn profile as a CV - Education	9
Figure 10	LinkedIn profile as a CV - Contact	9
Figure 11	LinkedIn profile as a CV - Top Skills	9
Figure 12	LinkedIn profile as a CV - Languages	9
Figure 13	LinkedIn profile as a CV - Certifications	10
Figure 14	LinkedIn profile as a CV - Honors-Awards	10
Figure 15	LinkedIn profile as a CV - Publications	10
Figure 16	LinkedIn profile as a CV - Patents	10
Figure 17	LinkedIn profile as a CV - Personal Information	10
Figure 18	Curriculum Lattes - Personal Information	11
Figure 19	Curriculum Lattes - Address and Contact	12
Figure 20	Curriculum Lattes - Academic Education	13
Figure 21	Curriculum Lattes - Professional Occupation	13
Figure 22	Curriculum Lattes - Occupation Area	14
Figure 23	CIÊNCIAVITAE - Identification: Personal Information (part 1)	14
Figure 24	CIÊNCIAVITAE - Identification: Personal Information (part 2)	15
Figure 25	CIÊNCIAVITAE - Identification: Personal Information (part 3)	15
Figure 26	CIÊNCIAVITAE - Identification: Contacts (part 1)	15
Figure 27	CIÊNCIAVITAE - Identification: Contacts (part 2)	16
Figure 28	CIÊNCIAVITAE - Identification: Contacts (part 3)	16
Figure 29	CIÊNCIAVITAE - Identification: Contacts (part 4)	17
Figure 30	CIÊNCIAVITAE - Identification: Acting Domains	17
Figure 31	CIÊNCIAVITAE - Identification: Languages	18
Figure 32	CIÊNCIAVITAE - Identification: Summary	18
Figure 33	CIÊNCIAVITAE - Education (part 1)	19

Figure 34	CIÊNCIAVITAE - Education (part 2)	19
Figure 35	CIÊNCIAVITAE - Education (part 3)	20
Figure 36	CIÊNCIAVITAE - Professional Path (part 1)	20
Figure 37	CIÊNCIAVITAE - Professional Path (part 2)	20
Figure 38	CIÊNCIAVITAE - Projects (part 1)	21
Figure 39	CIÊNCIAVITAE - Projects (part 2)	21
Figure 40	CIÊNCIAVITAE - Projects (part 3)	22
Figure 41	CIÊNCIAVITAE - Projects (part 4)	22
Figure 42	CIÊNCIAVITAE - Productions (part 1)	23
Figure 43	CIÊNCIAVITAE - Productions (part 2)	23
Figure 44	CIÊNCIAVITAE - Productions (part 3)	24
Figure 45	CIÊNCIAVITAE - Activities	24
Figure 46	CIÊNCIAVITAE - Distinctions (part 1)	25
Figure 47	CIÊNCIAVITAE - Distinctions (part 2)	25
Figure 48	Syone's internal CV - Personal Information	26
Figure 49	Syone's internal CV - Executive Summary	26
Figure 50	Syone's internal CV - Professional Experience	26
Figure 51	Syone's internal CV - Education	27
Figure 52	Syone's internal CV - Languages	27
Figure 53	Syone's internal CV - Technologies	28
Figure 54	Entities found in the text	32
Figure 55	Relationships found in the text	32
Figure 56	Information Extraction workflow	33
Figure 57	Text sample and its named entities	37
Figure 58	More complex text sample and its named entities	37
Figure 59	Text sample and its dependency parse tree	38
Figure 60	Jaccard Similarity	39
Figure 61	One-hot encoding	41
Figure 62	Skip-gram	41
Figure 63	CBOW	42
Figure 64	Transformer's architecture	43
Figure 65	BERT's architecture	43
Figure 66	USE's input and output paradigm	44
Figure 67	Cosine similarity	46
Figure 68	General architecture of the system	48
Figure 69	Architecture of approach A	49
Figure 70	Architecture of approach B	50
Figure 71	Text extractor and XML extractor engines' class diagram	53

Figure 72	Example of an Education section	54
Figure 73	Separator of each Professional Experience section's entry	55
Figure 74	Separator of each Education section's entry	55
Figure 75	Example of a Professional Experience entry without the "Software and Environments used" field	56
Figure 76	Example of a Professional Experience entry without the "Occupation or position held" field	56
Figure 77	Example of an Education entry without the "Principal subjects / occupational skills covered" field	57
Figure 78	Example of an Education entry with multiple "Title of qualification awarded" fields	57
Figure 79	Example of an Education entry without the "Title of qualification awarded" field	58
Figure 80	Candidate with an invalid date format	59
Figure 81	Candidate with a date format from which it was not possible to know whether it was a start date or an end date	59
Figure 82	Candidate using "to" to separate the start date from the end date	60
Figure 83	Candidate using "till" to separate the start date from the end date	60
Figure 84	Knowledge extractor engine's class diagram (part 1)	63
Figure 85	Knowledge extractor engine's class diagram (part 2)	63
Figure 86	Knowledge extractor engine's class diagram (part 3)	63
Figure 87	Document builder engine's class diagram	64
Figure 88	Graph of math function that returns score based on number of years	81

---

## LIST OF TABLES

---

Table 1	Comparison between the algorithms	44
Table 2	CVs chosen for each job offer based on the application of the algorithms	67
Table 3	Intersection of the top 5 CVs chosen by the various algorithms for the Backend Developer GO job offer	67
Table 4	Intersection of the top 5 CVs chosen by the various algorithms for the Backend Developer job offer	68
Table 5	Intersection of the top 5 CVs chosen by the various algorithms for the BE Developer job offer	68
Table 6	Intersection of the top 5 CVs chosen by the various algorithms for the Business Development Manager job offer	68
Table 7	Intersection of the top 5 CVs chosen by the various algorithms for the Frontend Developer (Angular) job offer	68
Table 8	Intersection of the top 5 CVs chosen by the various algorithms for the Frontend Developer (Vue.js) (1) job offer	69
Table 9	Intersection of the top 5 CVs chosen by the various algorithms for the Frontend Developer (Vue.js) (2) job offer	69
Table 10	Intersection of the top 5 CVs chosen by the various algorithms for the Infrastructure Architect and Pre-Sales job offer	69
Table 11	Intersection of the top 5 CVs chosen by the various algorithms for the IT Recruiter job offer	69
Table 12	Intersection of the top 5 CVs chosen by the various algorithms for the Junior Developers (1) job offer	70
Table 13	Intersection of the top 5 CVs chosen by the various algorithms for the ML/Data Engineer job offer	70
Table 14	Intersection of the top 5 CVs chosen by the various algorithms for the .NET Developer (1) job offer	70
Table 15	Intersection of the top 5 CVs chosen by the various algorithms for the .NET Developer (2) job offer	70
Table 16	Intersection of the top 5 CVs chosen by the various algorithms for the QA Engineer job offer	71
Table 17	Intersection of the top 5 CVs chosen by the various algorithms for the Scrum Master (1) job offer	71

Table 18	Intersection of the top 5 CVs chosen by the various algorithms for the Scrum Master (2) job offer	71
Table 19	Intersection of the top 5 CVs chosen by the various algorithms for the Solution Designer - Hardware job offer	72
Table 20	Intersection of the top 5 CVs chosen by the various algorithms for the Junior Developers (2) job offer	72
Table 21	Intersection of the top 5 CVs chosen by the various algorithms for the System Administrator job offer	72
Table 22	Intersection of the top 5 CVs chosen by the various algorithms for the Tech Lead/Senior Frontend job offer	72
Table 23	CVs chosen for each job offer based on the application of the all-mpnet-base-v2 model	73
Table 24	CVs chosen for each job offer based on the application of the all-MiniLM-L6-v2 model	74
Table 25	CVs chosen for each job offer based on the application of USE with Euclidean distance	75
Table 26	CVs chosen for each job offer based on the application of USE with Manhattan distance	76
Table 27	CVs chosen for each job offer based on the application of USE with Chebyshev distance	77
Table 28	CVs chosen for each job offer based on the application of TF-IDF to the preprocessed sample	78
Table 29	CVs chosen for each job offer based on the application of USE to the preprocessed sample	79
Table 30	Top 5 candidates for the job offer presented in Appendix A	88

---

## LIST OF LISTINGS

---

2.1	Description of the ontology with no instances . . . . .	28
2.2	Description of the ontology with instances . . . . .	29
2.3	Import library, load English model and create Doc object . . . . .	34
2.4	Sentence splitting . . . . .	35
2.5	Tokenization . . . . .	35
2.6	POS tagging . . . . .	35
2.7	Lemmatization . . . . .	36
2.8	Named Entity Recognition . . . . .	37
2.9	Dependency parsing . . . . .	37
5.1	Import libraries, modules and functions . . . . .	65
5.2	Load the USE's TF Hub module . . . . .	65
5.3	General code approach to apply the algorithm and the similarity measure (in this case, USE with cosine similarity) . . . . .	66
6.1	Dictionary of settings used as an example . . . . .	88
C.1	JSON file corresponding to the sample job offer . . . . .	108
D.1	JSON file corresponding to the sample CV . . . . .	112

---

## ACRONYMS

---

### A

AI Artificial Intelligence.

### B

BERT Bidirectional Encoder Representations from Transformers.

### C

CBOW Continuous Bag Of Words.

CV *Curriculum Vitae*.

### D

DAN Deep Averaging Network.

DNN Deep Neural Network.

### E

EU European Union.

### G

GUI Graphical User Interface.

### H

HR Human Resources.

### I

IE Information Extraction.

IT Information Technology.

### J

JSON JavaScript Object Notation.

## K

KE Knowledge Engineering.

## M

ML Machine Learning.

## N

NER Named Entity Recognition.

NLP Natural Language Processing.

## P

POS Part-Of-Speech.

PV-DBOW Distributed Bag Of Words version of Paragraph Vector.

PV-DM Distributed Memory version of Paragraph Vector.

## T

TF-IDF Term Frequency — Inverse Document Frequency.

## U

UPOS Universal Part-Of-Speech tag set.

URL Uniform Resource Locator.

USE Universal Sentence Encoder.



---

## INTRODUCTION

---

In this first chapter of the dissertation, the Master's Project is introduced along with its context and motivation. Furthermore, the necessary objectives to complete it are also detailed, the research hypothesis and the corresponding research methodology are specified and the document structure is presented.

### 1.1 CONTEXT

Recruitment consists in hiring the best candidates for a given job offer according to their skills and the first task of this process is to screen the CVs of all the job applicants. However, this can be a very exhausting and tedious chore for recruiters, considering they have to manually narrow down the most appropriate candidates from an immensely large pool. (Amin et al., 2019)

As a proof, the screening process can be condensed as stated bellow: (Catherine et al., 2010)

1. Understand the job offer's requirements in terms of the skills that are mandatory (such as certain technologies) and those that are optional, but preferable (for example, speaking a particular language), the experience criteria if any, the preference for the location of the candidate, *etc.*;
2. Go through each and every of the applications and discard those which do not satisfy the skills required for the job;
3. Out of the remaining candidates, find the best match for the job. To do this, the recruiter has to read the CVs in detail and compare them with the job offer. In addition, since the number of candidates who can be interviewed is limited, the recruiter has to make a relative judgement on them.

Therefore, when using such a sequential and relative selection procedure, manual screening may miss potentially better candidates for the job opening which, in turn, may result in a suboptimal pool of potential employees. (Mehta et al., 2013)

Moreover, with less than 5% of people being selected from these applications, it is impractical for the recruiters to go through the CVs one by one for these limited number of openings. (Daryani et al., 2020)

Another problem faced by recruiters is the fact that the CVs do not follow any specific pattern, *i.e.*, beyond the fact that the CVs are in various formats, such as .pdf, .docx, .jpg, *etc.*, they do not have a standard style of presenting their content. As an attempt to solve this problem, many job portals decided to provide an online form so that the job applicants could fill up all the information of their CV in a structured manner, creating the so called "candidate metadata". Nevertheless, this solution requires redundant efforts from the candidates, which often leads them to fill that template with incomplete information. (Daryani et al., 2020)

## 1.2 MOTIVATION

In the past few years, there have been developed some automatic CV screening tools, in order to automate that recruitment process task. However, most of them show vulnerabilities: either in precision, the lack of empirical validation or even the presentation of bias (ending up discriminating candidates).

Enachescu (2019), for instance, states some negative impacts on the accuracy of the system. On the other hand, Laumer and Eckhardt (2009), for example, did not validate their conceptual approach with real unstructured data. Lastly, the American multinational technology company, Amazon, *e.g.*, developed an Artificial Intelligence (AI) recruiting tool that was not rating candidates for software developer jobs and other technical posts in a gender-neutral way (ama).

This Master's Work entails the creation of a system that will overcome those issues, never forgetting the principal aim: being capable of screening CVs automatically.

## 1.3 OBJECTIVES

The main objective of this Master's Work is to identify which applicants are most suitable for a specific job offer. To attain this objective, the steps below will be followed:

- Extract crucial information about the candidates and the job offers;
- Apply a function capable of calculating the similarity between a CV and a job offer.

## 1.4 RESEARCH HYPOTHESIS

The research hypothesis that will be proved with this Master's Work is stated bellow:

“It is possible to develop an algorithm capable of automatically matching the most suitable applicants to a certain job offer”.

## 1.5 RESEARCH METHODOLOGY

In order to accomplish this Master’s Work, an iterative methodology based on literature review, solution proposal, implementation and testing will be carried on. To achieve this approach, the following steps will be performed:

- Bibliographic study to deeply understand the state of the art in CV sources and formats, Information Extraction (IE) components and steps, Natural Language Processing algorithms and similarity metrics;
- Synthesis and description of the products of that study;
- Analysis of the sample of CVs and job offers that will be used as a case study;
- Extraction of the important information from those CVs and job offers;
- Implementation of an algorithm where the recruiters will be able to choose the job offer and the respective candidates and get as a result the  $x$  best candidates that applied for that job, according to the number of applicants they are willing to interview;
- Testing of the developed algorithm and evaluation and discussion of the results.

After the pursuance of these steps, if the results are still not favorable, then iterate all over again.

## 1.6 DOCUMENT STRUCTURE

This document is composed of seven different chapters with the following contents:

1. Introduction - In the first chapter the problem is contextualized and the motivation to proceed with its resolution is exposed. In addition, the main goals of this work are detailed and the research hypothesis to be proved is stated. Lastly, the research methodology containing the steps to be followed is proposed;
2. State of the art - In this chapter the concepts of CV sources and formats, semantic information extraction, similarity algorithms and similarity measures are studied and the results of that study are presented;
3. Proposed approach - In the third chapter, a solution to the problem is proposed, which, in turn, is illustrated with a diagram for better understanding;

4. CVs processing - In this chapter the engines responsible for processing the CVs and the challenges that arose with their development are explained;
5. Similarity algorithm (approach A) - In the fifth chapter the general coding strategy for calculating the affinity between job offers and CVs in the first similarity algorithm approach is presented, along with the elucidation of the various techniques applied to achieve that affinity and their respective results and conclusions;
6. Similarity algorithm (approach B) - In this chapter the second similarity algorithm approach that was followed to calculate the affinity between job offers and CVs is explained and the results and conclusions of that approach are presented;
7. Conclusion - The last chapter contains a recap of all the work done throughout this Master's Project, the obstacles that emerged during the development of the project, the conclusions obtained and the future work that could be done in the following iterations.

---

## STATE OF THE ART

---

Before proposing an approach to solve the problem under discussion, it is important to analyze the current state of the art regarding the scope of this Master's Work. Therefore, in this second chapter, the concepts of CV sources and formats, semantic information extraction, similarity algorithms and similarity measures are explored.

### 2.1 CV SOURCES AND FORMATS

A *Curriculum Vitae* (Latin for "course of life"), as the name implies, is a document containing the information about an individual's career, education and skills and it is usually used when they want to apply to a certain job offer. However, there is still no standard format globally used and accepted at the moment and, as a result, there are several different types of CVs, according to the area of expertise or country.

Despite that, there is a set of patterns that can be seen in the various types of CVs. Therefore, in this section, the Europass ([Eur](#)), LinkedIn profile as a CV ([Lin](#)), Curriculum Lattes ([Lat](#)), CIÊNCIAVITAE ([Cie](#)) and Syone's internal CV formats will be presented and analyzed and, after that, an ontology supporting all the information it will be necessary to extract from each CV will be developed.

#### 2.1.1 *Europass*

The Europass was created by the Directorate General for Education and Culture of the European Union so that citizens could follow a common pattern among all the countries of the European Union (EU). It is a very complete format and can be divided into five main sections: Personal Information, Work Experience, Education and Training, Personal Skills and Additional Information.

The first section, Personal Information (Figure 1), contains the individual's name, picture, contact information (such as home address, telephone/mobile number, e-mail address, websites and messaging accounts), sex, date of birth and nationalities. It also contains the description of the job position they have applied for.







<p>PERSONAL INFORMATION</p> <div style="background-color: #ccc; width: 100px; height: 80px; margin: 10px 0;"></div> <p>JOB APPLIED FOR POSITION PREFERRED JOB STUDIES APPLIED FOR</p>	<p><b>Replace with First name(s) Surname(s)</b>  <small>[All CV headings are optional. Remove any empty headings]</small></p> <p> Replace with house number, street name, city, postcode, country</p> <p> Replace with telephone number  Replace with mobile number</p> <p> <u>State e-mail address</u></p> <p> <u>State personal website(s)</u></p> <p> <u>Replace with type of IM service</u> Replace with messaging account(s)</p> <p><small>Sex Enter sex   Date of birth dd/mm/yyyy   Nationality Enter nationality/-ies</small></p> <p><b>Replace with preferred job / job applied for / studies applied for / position (delete non relevant headings in left column)</b></p>
---	---

Figure 1: Europass - Personal Information

The Work Experience section (Figure 2) holds the previous occupations the individual has had in the past and the respective period of time, company and location in which they had performed them enumerated chronologically. In addition, the responsibilities the person had and the sector where they worked are detailed.

<p>WORK EXPERIENCE</p> <p>Replace with dates (from - to)</p>	<hr style="border: 1px solid #0070c0; margin-bottom: 5px;"/> <p><small>[Add separate entries for each experience. Start from the most recent.]</small></p> <p><b>Replace with occupation or position held</b></p> <p>Replace with employer's name and locality (if relevant, full address and website)</p> <ul style="list-style-type: none"> <li>▪ Replace with main activities and responsibilities</li> </ul> <p><small>Business or sector</small> Replace with type of business or sector</p>
--	---

Figure 2: Europass - Work Experience

The Education and Training section (Figure 3) is very similar to the previous one. It contains the individual's qualifications (and corresponding European Qualification Framework (or other) levels) and, once again, the respective period of time, organization and location in which they achieved them enumerated chronologically. Moreover, the main subjects covered are also described.

<p>EDUCATION AND TRAINING</p> <p>Replace with dates (from - to)</p>	<hr style="border: 1px solid #0070c0; margin-bottom: 5px;"/> <p><small>[Add separate entries for each course. Start from the most recent.]</small></p> <p><b>Replace with qualification awarded</b></p> <p style="text-align: right;"><small>Replace with European Qualification Framework (or other) level if relevant</small></p> <p>Replace with education or training organisation's name and locality (if relevant, country)</p> <ul style="list-style-type: none"> <li>▪ Replace with a list of principal subjects covered or skills acquired</li> </ul>
---	--

Figure 3: Europass - Education and Training

The next section, Personal Skills (Figure 4 and Figure 5), starts with the individual’s language skills and the respective proficiency level regarding understanding, speaking and writing. Then, it also displays the individual’s communication skills, organisational/managerial skills, job-related skills, computer skills and other skills. In the end, the person can also state whether they have a driving license and, if so, its category.

**PERSONAL SKILLS** ■

[Remove any headings left empty.]

**Mother tongue(s)** Replace with mother tongue(s)

**Other language(s)**

	UNDERSTANDING		SPEAKING		WRITING
	Listening	Reading	Spoken interaction	Spoken production	
Replace with language	Enter level	Enter level	Enter level	Enter level	Enter level
	Replace with name of language certificate. Enter level if known.				
Replace with language	Enter level	Enter level	Enter level	Enter level	Enter level
	Replace with name of language certificate. Enter level if known.				

Levels: A1/2: Basic user - B1/2: Independent user - C1/2 Proficient user  
Common European Framework of Reference for Languages

**Communication skills** Replace with your communication skills. Specify in what context they were acquired. Example:  

- good communication skills gained through my experience as sales manager

**Organisational / managerial skills** Replace with your organisational / managerial skills. Specify in what context they were acquired.  
 Example:  

- leadership (currently responsible for a team of 10 people)

**Figure 4: Europass - Personal Skills (part 1)**

**Job-related skills** Replace with any job-related skills not listed elsewhere. Specify in what context they were acquired.  
 Example:  

- good command of quality control processes (currently responsible for quality audit)

**Computer skills** Replace with your computer skills. Specify in what context they were acquired. Example:  

- good command of Microsoft Office™ tools

**Other skills** Replace with other relevant skills not already mentioned. Specify in what context they were acquired.  
 Example:  

- carpentry

**Driving licence** Replace with driving licence category/-ies. Example:  

- B

Figure 5: Europass - Personal Skills (part 2)

Finally, the last section, Additional Information (Figure 6), has any other information about the individual that was not stated above and that they might consider relevant to mention, such as publications, presentations, projects, among others.

## ADDITIONAL INFORMATION

Publications	Replace with relevant publications, presentations, projects, conferences, seminars, honours and awards, memberships, references. Remove headings not relevant in the left column.
Presentations	
Projects	Example of publication:
Conferences	▪ How to write a successful CV, New Associated Publishers, London, 2002.
Seminars	Example of project:
Honours and awards	▪ Devon new public library. Principal architect in charge of design, production, bidding and construction supervision (2008-2012).
Memberships	
References	

Figure 6: Europass - Additional Information

## 2.1.2 LinkedIn profile as a CV

LinkedIn is the world's largest professional network on the Internet. This platform allows people to find job opportunities through their profile which showcases past professional experiences, skills, licenses and certifications, *etc.*. Currently, it is also possible to export a person's LinkedIn profile as a CV which, in turn, has ten main sections: Summary, Experience, Education, Contact, Top Skills, Languages, Certifications, Honors-Awards, Publications and Patents.

The Summary section (Figure 7) presents a brief summary about the person. That summary usually contains a description about their skills, achievements or previous job experiences.

### Summary

Currently, I am a student at University of Minho.

Figure 7: LinkedIn profile as a CV - Summary

The Experience section (Figure 8) contains the previous work experiences with the corresponding start and end dates. It also includes the company's name, the job role, the country where they took place and a brief description.

### Experience

Inditex

Salesperson

June 2018 - December 2019 (1 year 7 months)

Portugal

I was responsible for selling apparel.

Figure 8: LinkedIn profile as a CV - Experience



The Education section (Figure 9) contains the previous education miles which include the respective school, degree, field and start and end dates.

## Education

University of Minho

Masters, Software Engineering · (2017 - 2022)

Figure 9: LinkedIn profile as a CV - Education

The Contact section (Figure 10) contains the person's contact information, such as address, mobile number, email and website.

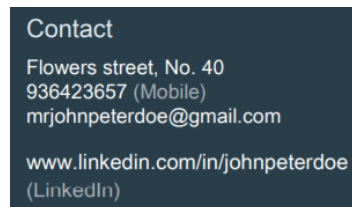


Figure 10: LinkedIn profile as a CV - Contact

The Top Skills section (Figure 11) contains the best skills of the individual.

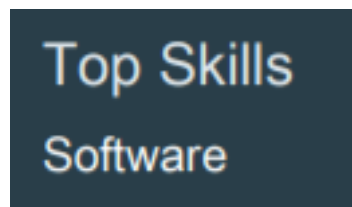


Figure 11: LinkedIn profile as a CV - Top Skills

The Languages section (Figure 12) contains the individual's language skills and the corresponding proficiency level.

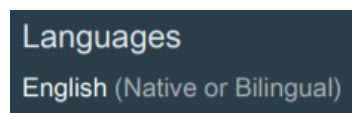


Figure 12: LinkedIn profile as a CV - Languages

In the Certifications section (Figure 13) the individual can state the respective certifications obtained.

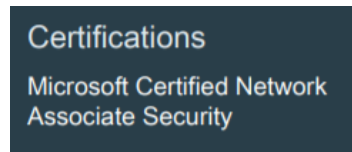


Figure 13: LinkedIn profile as a CV - Certifications

In the section Honors-Awards (Figure 14) the person can enumerate the honors and awards earned throughout their life.

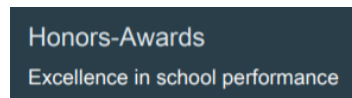


Figure 14: LinkedIn profile as a CV - Honors-Awards

The section Publications (Figure 15) contains the title of the publications the individual has produced.

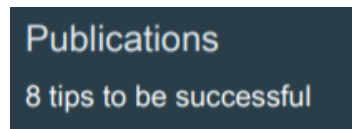


Figure 15: LinkedIn profile as a CV - Publications

In the section Patents (Figure 16) the person can list all the patents they detain.

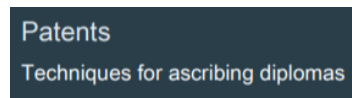


Figure 16: LinkedIn profile as a CV - Patents

Lastly, there is one more section with no title which contains the personal information, these being the name of the person, a brief description and the current location (Figure 17).

**John Doe**  
 Student at University of Minho  
 Braga

Figure 17: LinkedIn profile as a CV - Personal Information

### 2.1.3 Curriculum Lattes

The Curriculum Lattes was created by the National Council for Scientific and Technological Development to meet the need to bring together in a unique format the academic journey of students, professors and researchers of higher education in Brazil. Like the Europass, it can also be divided into five key sections: Personal Information, Address and Contact, Academic Education, Professional Occupation and Occupation Area.

The Personal Information section (Figure 18) contains the person's picture, name, date of birth, nationality, sex and colour or race. If the person does not have an Individual Registration, the field with the corresponding number can be left blank. However, if the person does have an Individual Registration, they not only have to fill that field, but they can also choose whether they want to use a social name or not (and, if so, supply that name). It also contains the ID and the respective issuing agency, Federative Unit and issue date. Additionally, it includes the passport number and the corresponding expiration date, issue date and issuing country. Lastly, it comprehends the parents' names.

**Informação pessoal**

Foto de perfil

**Nome civil**

Primeiro nome  
Informe seu primeiro nome ex: "José"

Sobrenome  
Informe seu sobrenome completo ex: "Pereira da Silva Aquino"

**Dados pessoais**

Data de nascimento  
Informe sua data de nascimento

País de nascimento  
Selecione seu país de nascimento

Sexo  
Selecione o seu sexo

Cor ou Raça  
Informe sua cor ou raça

Você possui CPF?  
Receita Federal do Brasil

Número do CPF  
Informe seu CPF (apenas os números)

Número de identidade  
Informe o número de seu documento

Órgão emissor  
Informe o órgão emissor

UF  
Unidade

Data de emissão  
Informe a data de emissão

Número do passaporte  
Informe nº do seu passaporte

Data de validade  
Informe a data de validade do passaporte

Data de emissão  
Informe a data de emissão do passaporte

País emissor  
Informe o país onde foi expedido

Primeiro nome do pai  
Informe o primeiro nome do seu pai

Sobrenome do pai (nome de família)  
Informe o sobrenome completo do seu pai

Primeiro nome da mãe  
Informe o primeiro nome da sua mãe

Sobrenome da mãe (nome de família)  
Informe o sobrenome completo da sua mãe

Cancelar Próxima

Figure 18: Curriculum Lattes - Personal Information

In the second section, Address and Contact (Figure 19), the person can complete the information about their home and/or professional address. In the first option, the individual starts by filling their address information (such as country, zip code, street address, district, city and state) and, finally, their telephone and mobile numbers. In the latter option, the person has to fill the same information and one additional field where they state their institution.

Figure 19: Curriculum Lattes - Address and Contact

The next section, Academic Education (Figure 20), can be divided into two subsections: the completed academic education and the academic education in progress.

In the first subsection, if the actual completed academic education is elementary or high school, then the individual just needs to fill the name of the institution, the start year and the end year. Otherwise, if the actual completed academic education is Bachelor Degree, Master Degree or Doctorate Degree, then the person also has to fill the course, the title of the final document produced and the respective name of the supervisor. Additionally, they also need to state whether they had a scholarship and, if so, the funding agency.

The second subsection is very similar. For elementary and high school, the fields to be filled are the same, except the end year (as expected). For Bachelor Degree, Master Degree and Doctorate Degree, the fields are also the same except the end year, the final document information and the scholarship information.

**Formação acadêmica**

Formação acadêmica concluída

Instituição (nome da Instituição) Início (ano) Conclusão (ano)

Formação acadêmica em andamento

Instituição (nome da Instituição) Início (ano)

Cancelar Anterior Próxima

Figure 20: Curriculum Lattes - Academic Education

The Professional Occupation section (Figure 21) starts by asking if the person is currently working. If not, this section can be left blank. Otherwise, the individual needs to state in what institution they are working (the country field will be automatically filled), the type of bond, the role and the start year.

**Atuação profissional**

Alguma atuação profissional no momento?

Sim Não

Instituição / Universidade País

Tipo do vínculo Cargo

Desde (Ano)

Cancelar Anterior Próxima

Figure 21: Curriculum Lattes - Professional Occupation

The last section, Occupation Area (Figure 22), contains not only the actual occupation area, but also the individual's language skills and the respective proficiency level regarding understanding, reading, speaking and writing.

**Área de atuação**

Habilidades linguísticas  
Informe os idiomas e o nível de cada idioma selecionado

idioma	compreende	lê	fala	escreve
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Figure 22: Curriculum Lattes - Occupation Area

#### 2.1.4 CIÊNCIAVITAE

The CIÊNCIAVITAE was created by Foundation for Science and Technology and is directed to Portuguese or foreign people who work in the academic and research context in Portugal. One of the main reasons that led to its creation was the need to have a single curriculum shared by the different stakeholders in the national academic-scientific system. This format can be divided into seven principal sections: Identification, Education, Professional Path, Projects, Productions, Activities and Distinctions.

The first section, Identification, has five subsections: Personal Information, Contacts, Acting Domains, Languages and Summary.

The first subsection, Personal Information (Figure 23, Figure 24 and Figure 25), contains the person's picture, name, date of birth and gender. It also contains the person's citation name, the type of the author and their identifier.

### Informação pessoal

**ANEXAR FOTO**

📄 O tamanho de foto recomendado é de 123x123 pixels nos formatos JPEG ou PNG

**Nome completo**

**Data de nascimento**

**Gênero**

Feminino  Masculino

Figure 23: CIÊNCIAVITAE - Identification: Personal Information (part 1)

## Informação pessoal

### Nomes de citação

Nome de citação \*

Figure 24: CIÊNCIAVITAE - Identification: Personal Information (part 2)

## Informação pessoal

### Identificadores de autor

Tipo \*

Identificador \*

Figure 25: CIÊNCIAVITAE - Identification: Personal Information (part 3)

In the second subsection, Contacts (Figure 26, Figure 27, Figure 28 and Figure 29), the person needs to choose whether the email is personal or professional and declare the actual email. Then, the person also needs to choose whether the telephone is personal or professional and select the device, supply the indicative, the number and the extension. Next and once again, the individual needs to choose whether the address is personal or professional, write the actual address, the zip code, the location, the county and the country. Lastly, the individual has to choose the type of website and provide the Uniform Resource Locator (URL).

## Contactos

### Endereços de correio eletrónico

Utilização \*

Endereço de correio eletrónico \*

 Ex: endereco@mail.pt

Figure 26: CIÊNCIAVITAE - Identification: Contacts (part 1)

## Contactos

### Telefones

Utilização \*

Dispositivo \*

Indicativo

Número \*

Extensão

Figure 27: CIÊNCIAVITAE - Identification: Contacts (part 2)

## Contactos

### Moradas

Utilização \*

Endereço \*

Código postal \*

 Ex: 1249-074

Localidade \*

Concelho \*

País \*

Figure 28: CIÊNCIAVITAE - Identification: Contacts (part 3)



## Contactos

### Websites

Tipo de website \*

URL \*

Figure 29: CIÊNCIAVITAE - Identification: Contacts (part 4)

The subsection Acting Domains (Figure 30) contains the knowledge area, the topic and the keywords.

### Domínios de atuação

Área do conhecimento \* 

Tópico

Palavras-chave 

 Introduza novas palavras-chave separadas por ";" ou selecione uma já introduzida.

Figure 30: CIÊNCIAVITAE - Identification: Acting Domains

In the subsection Languages (Figure 31) the individual needs to choose the language, whether it is their mother tongue and the proficiency levels regarding speaking, reading, writing, comprehension and peer-review.

## Idiomas

Idioma \*

Selecione uma opção ▼

Idioma materno?

Conversaço

Selecione uma opção ▼

Leitura

Selecione uma opção ▼

Escrita

Selecione uma opção ▼

Compreens

Selecione uma opção ▼

Peer-review


Selecione uma opção ▼

Figure 31: CIÊNCIAVITAE - Identification: Languages

In the last subsection, Summary (Figure 32), the person can provide a brief description about themselves.

## Resumo

|

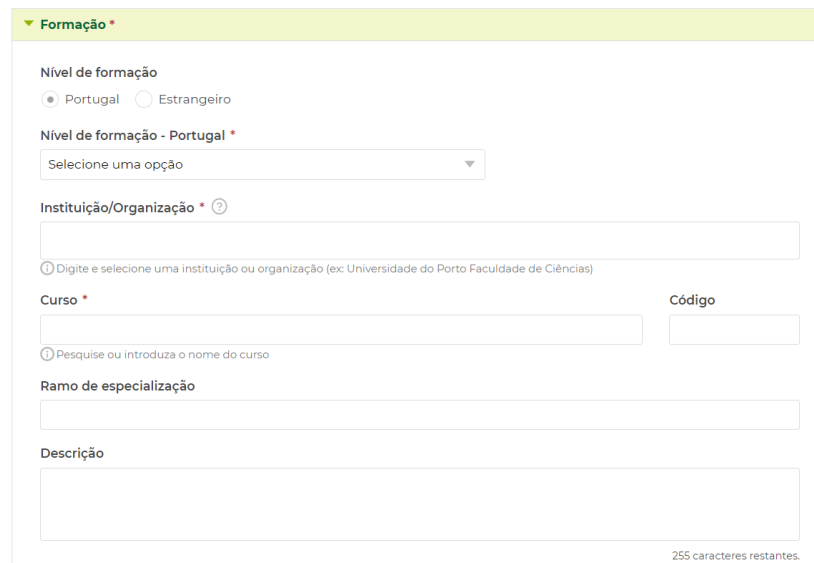


4000 caracteres restantes.

Figure 32: CIÊNCIAVITAE - Identification: Summary

In the next section, Education (Figure 33, Figure 34 and Figure 35), the individual starts by stating whether the education mile was obtained in Portugal or abroad. Then, they have to choose the level of education and provide the organization, course and corresponding code, specialization, description, grade, status and start and end dates. Then, the person can also supply the dissertation title, its ID, the organization where the thesis was defended and the information about the supervisors (name and role). Lastly, the knowledge area can also be included.

### Formação




**Formação \***

Nível de formação

Portugal  Estrangeiro

Nível de formação - Portugal \*

Selecione uma opção

Instituição/Organização \* 

① Digite e selecione uma instituição ou organização (ex: Universidade do Porto Faculdade de Ciências)

Curso \* Código

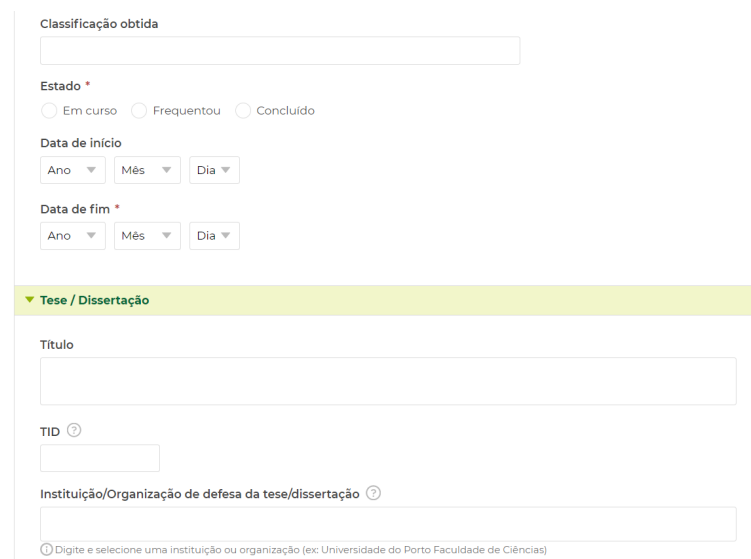
① Pesquise ou introduza o nome do curso

Ramo de especialização

Descrição

255 caracteres restantes.

Figure 33: CIÊNCIAVITAE - Education (part 1)



Classificação obtida

Estado \*

Em curso  Frequentou  Concluído

Data de início


Ano ▼ Mês ▼ Dia ▼


Data de fim \*

Ano ▼ Mês ▼ Dia ▼

**Tese / Dissertação**

Título

TID 

Instituição/Organização de defesa da tese/dissertação 

① Digite e selecione uma instituição ou organização (ex: Universidade do Porto Faculdade de Ciências)

Figure 34: CIÊNCIAVITAE - Education (part 2)

**Orientadores**

Nome	Papel desempenhado
<input type="text"/>	Selecione uma opção <span>▼</span> <span>🗑️</span>

**ADICIONAR**

**Áreas do conhecimento**

Áreas do conhecimento ?

📌 Digite e selecione uma área do conhecimento (ex. Ciências Sociais - Psicologia)

Figure 35: CIÊNCIAVITAE - Education (part 3)

The next section, Professional Path (Figure 36 and Figure 37), starts with the type of job, employer, professional bond, professional category, host institution and start and end dates. Moreover, it has the corresponding activities that can be associated with the Activities section.

### Percurso profissional 📄

**Informação geral \***

**Tipo \***

Selecione uma opção ▼

**Empregador \* ?**

📌 Digite e selecione uma instituição ou organização (ex: Universidade do Porto Faculdade de Ciências)

**Vínculo profissional \***

Selecione uma opção ▼

**Categoria profissional \***

Selecione uma opção ▼

**Instituição/Organização de acolhimento ?**

📌 Digite e selecione uma instituição ou organização (ex: Universidade do Porto Faculdade de Ciências)

**Data de início \***

Ano ▼ Mês ▼ Dia ▼

**Data de fim**

Ano ▼ Mês ▼ Dia ▼

📌 Deixar em branco se atual

**Atividades associadas**

Sem informação, por favor adicione um novo registro.

**CRIAR** **ASSOCIAR**

Figure 36: CIÊNCIAVITAE - Professional Path (part 1)


**Atividades associadas**

Sem informação, por favor adicione um novo registro.

**CRIAR** **ASSOCIAR**

Figure 37: CIÊNCIAVITAE - Professional Path (part 2)

The Projects section (Figure 38, Figure 39, Figure 40 and Figure 41) contains the function and the dates between which they were performed. Moreover, it contains the project's title, the description, the type of financing, the start and end dates, the project's status and the organization. It also has the information about the financing itself, such as the financing entity, the financing program and the year of assignment. Finally, it has the project's references (including the reference code, the URL and the type of relation), the information about other participants, the knowledge areas and the keywords.

**Projetos** 

**▼ Tipo de participação \***

Função desempenhada


Data de início da participação \*


Ano  Mês  Dia

Data de fim da participação

Ano  Mês  Dia

**▼ Informação de projeto \***

Título do projeto \* 

  
 Pesquise ou introduza o título do projeto

Descrição

4000 caracteres restantes

Figure 38: CIÊNCIAVITAE - Projects (part 1)

Tipo de financiamento \*

Data de início


Ano  Mês  Dia


Data de fim

Ano  Mês  Dia


Estado do projeto


Concluído  Desativado  Em curso

Instituição/Organização/Centro de I&D 

  
 Digite e selecione uma instituição ou organização (ex: Universidade do Porto Faculdade de Ciências)

**▼ Financiamento**

Entidade financiadora 

  
 Digite e selecione uma entidade financiadora (ex: FCT)

Programa de financiamento

Ano da atribuição

Ano

Figure 39: CIÊNCIAVITAE - Projects (part 2)

**Referências do projeto \***

Referência \*      URL      Tipo de relação \*

Código de referência do projeto      http(s)://       Parte de       Próprio

ADICIONAR

---

**Responsáveis e outros participantes**

Outros participantes

ADICIONAR

---

**Áreas do conhecimento e palavras-chave**

Áreas do conhecimento ⓘ

ⓘ Digite e selecione uma área do conhecimento (ex. Ciências Sociais - Psicologia)

Palavras-chave ⓘ

ⓘ Introduza novas palavras-chave separadas por ";" ou selecione uma já introduzida.

Figure 40: CIÊNCIAVITAE - Projects (part 3)

Figure 41: CIÊNCIAVITAE - Projects (part 4)

The Productions section hosts every work the individual has produced and it varies according to the category and type chosen. In case of a publication in a journal (Figure 42, Figure 43 and Figure 44), for instance, the individual has to provide the title of the article, the name of the journal, the section, the volume, the edition, the main and final pages, the publication date, the country and city of publication and the URL. Next, the person needs to supply the information about the identifier (the type, the actual identifier and the type of relation). Then, they also need to state the name to cite and whether it is theirs or from another author. Lastly, it has the corresponding projects that can be linked from the section above, the knowledge areas and the keywords.

**Produções** 

**Categoria \***

**Tipo \***

---

**▼ Detalhes \***

**Título do artigo \***

**Jornal de notícias**

**Secção**  **Volume**  **Edição**  **Página inicial**

**Página final**




**Data de publicação \***  
 Ano  Mês  Dia

**País de publicação**  **Cidade de publicação**

**URL**



Figure 42: CIÊNCIAVITAE - Productions (part 1)

**▼ Identificadores \***

Tipo de identificador *	Identificador *	Tipo de relação *	
<input type="text" value="ID Não-standard da fonte de infor"/>	<input type="text" value="cv-prod-id-2730341"/>	<input type="radio"/> Parte de <input checked="" type="radio"/> Próprio	
<input type="text" value="Selecione uma opção"/>	<input type="text"/>	<input type="radio"/> Parte de <input checked="" type="radio"/> Próprio	

---

**▼ Autores**

Nome de citação *	Próprio?	Autor correspondente?	
<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	

---

**▼ Projetos financiadores**

Sem informação, por favor adicione um novo registo.

Figure 43: CIÊNCIAVITAE - Productions (part 2)

Figure 44: CIÊNCIAVITAE - Productions (part 3)

Like the previous section, the Activities section shelters a lot of different details about the activities, according to the type chosen. In the case of tutoring (Figure 45), for example, the details involve the topic, the start and end dates, the name of the student and the description.

Figure 45: CIÊNCIAVITAE - Activities

The last section, Distinctions (Figure 46 and Figure 47), houses the distinctions' type, name and year of the attribution and the promoting entity. Additionally, it contains the country, the description and the expiration date (if applicable). Lastly, the person can also state the knowledge areas and the keywords.




## Distinções

**▼ Detalhes \***

Tipo \*

Nome da distinção \*

Ano da atribuição \*

Entidade(s) promotora(s) 


 Digite e selecione uma instituição ou organização (ex: Universidade do Porto Faculdade de Ciências)

Figure 46: CIÊNCIAVITAE - Distinctions (part 1)



**▼ Outra informação**

País

Descrição  
  
255 caracteres restantes.

Data de expiração (se aplicável)

**▼ Áreas do conhecimento e palavras-chave**

Áreas do conhecimento   
  
 Digite e selecione uma área do conhecimento (ex. Ciências Sociais - Psicologia)



Palavras-chave   
  
 Introduza novas palavras-chave separadas por "\*" ou selecione uma já introduzida.

Figure 47: CIÊNCIAVITAE - Distinctions (part 2)

### 2.1.5 Syone's internal CV

At the time of application, candidates to Syone must fill in the company's internal CV so that the Human Resources (HR) team has access to the most relevant information for the hiring process in one single format. This format has six key sections: Personal Information, Executive Summary, Professional Experience, Education, Languages and Technologies.

The first section, Personal Information (Figure 48), contains the individual's personal data, such as name, job description, age and gender.

## PERSONAL INFORMATION

**NAME:**

**JOB DESCRIPTION:**

**AGE:**

**GENDER:**

Figure 48: Syone's internal CV - Personal Information

In the second section, Executive Summary (Figure 49), the person can write a brief description about themselves.

## EXECUTIVE SUMMARY

Figure 49: Syone's internal CV - Executive Summary

The next section, Professional Experience (Figure 50), contains the individual's past work experiences. Therefore, it includes the respective period of time they lasted, the company's name, the type of business/sector, the role, the main activities and responsibilities and the software and environments used.

PROFESSIONAL EXPERIENCE	
Dates (from to)	
Company Name	
Type of business or sector	
Occupation or position held	
Main activities and responsibilities	
Software and Environments used	

Figure 50: Syone's internal CV - Professional Experience

The Education section (Figure 51), contains the person’s education and training miles. Consequently, it includes the start and end dates, the name and type of organization providing the qualification, the principal subjects/occupational skills covered and the actual qualification awarded.

**EDUCATION**

Dates (from to)	
Name and type of organization providing education and training	
Principal subjects / occupational skills covered	
Title of qualification awarded	

Figure 51: Syone’s internal CV - Education

The Languages section (Figure 52) contains a table with all the individual’s language skills and the corresponding proficiency levels regarding reading, writing and speaking.

**LANGUAGES**

Other Languages	Reading Skills	Writing Skills	Verbal Skills

Language: Basic | Average | Good | Very Good | Native

Figure 52: Syone’s internal CV - Languages

Finally, the Technologies section (Figure 53) has a table with all the person’s technology skills regarding programming languages, databases, operating systems and tools and the corresponding level, duration, start and end dates and description.

## TECHNOLOGIES

Technology	Level	Duration	Date	Detail Description
Programming Languages				
Databases				
Operating Systems				
Tools				

**Level:** Basic | Average | Good | Very Good | Advanced  
**Duration:** X Months | X Years  
**Date:** Year(s) you worked with the technology (e.g. 2015 – 2016; 2021)

Figure 53: Syone's internal CV - Technologies

### 2.1.6 Considerations

Within this topic, other CV formats were analyzed. The Career Center of the Academic Format of the University of Washington (*Aca*), for instance, was one of them, but, like Curriculum Lattes and CIÊNCIAVITAE, this one aims at people who work in the academic and research fields and, so, was not worth mentioning in this dissertation. Another CV format that was explored was the College Art Association's Visual Artist Format (*CAA*). Nevertheless, this is directed to people working in the art field, hence this format was also not highlighted, assuming that Syone will not be getting applications from these individuals. Finally, after an extensive research, no more CV formats in vogue were found.

### 2.1.7 Ontology definition

In order to decide all the information it will be necessary to extract from each CV, an ontology was created based on Subsection 2.1.5.

Below is the description of the ontology in OntoDL+ (*Ont*) with no instances and the same ontology with instances. Note that those instances were based on the CV of Subsection 2.1.2.

```

1 Ontologia cv
3 conceitos {
4     CurriculumVitae ,
5     PersonalInformation[name: string , jobDescription: string , age: string , gender
6     : string ],

```

```

7   ExecutiveSummary[description: string],
   ProfessionalExperience[startDate: string, endDate: string, company: string,
   typeOfBusiness: string, job: string, activities: string, technologies: string
   ],
   Education[startDate: string, endDate: string, organization: string, subjects:
   string, qualification: string],
9   Language[language: string, reading: string, writing: string, verbal: string],
   Technology[technology: string, level: string, startDate: string, endDate:
   string, description: string],
11  ProgrammingLanguage,
   Database,
13  OperatingSystem,
   Tool
15 }

17 relacoes {
   has
19 }

21 triplos {
   ProgrammingLanguage = isa => Technology;
23  Database = isa => Technology;
   OperatingSystem = isa => Technology;
25  Tool = isa => Technology;

27  CurriculumVitae = has => PersonalInformation;
   CurriculumVitae = has => ExecutiveSummary;
29  CurriculumVitae = has => ProfessionalExperience;
   CurriculumVitae = has => Education;
31  CurriculumVitae = has => Language;
   CurriculumVitae = has => Technology
33 }.

```

Listing 2.1: Description of the ontology with no instances

```

1  Ontologia cv
3  conceitos {
   CurriculumVitae,
5   PersonalInformation[name: string, jobDescription: string, age: string, gender
   : string],
   ExecutiveSummary[description: string],
7   ProfessionalExperience[startDate: string, endDate: string, company: string,
   typeOfBusiness: string, job: string, activities: string, technologies: string
   ],
   Education[startDate: string, endDate: string, organization: string, subjects:
   string, qualification: string],

```

```

9   Language[language: string , reading: string , writing: string , verbal: string],
   Technology[technology: string , level: string , startDate: string , endDate:
11  string , description: string],
   ProgrammingLanguage ,
   Database ,
13  OperatingSystem ,
   Tool
15 }

17 individuos {
   personalinformation ,
19  executivesummary ,
   professionalexperience ,
21  education ,
   language ,
23  programminglanguage ,
   database ,
25  operatingsystem ,
   tool
27 }

29 relacoes {
   has
31 }

33 triplos {
   ProgrammingLanguage = isa => Technology;
35  Database = isa => Technology;
   OperatingSystem = isa => Technology;
37  Tool = isa => Technology;

39  CurriculumVitae = has => PersonalInformation;
   CurriculumVitae = has => ExecutiveSummary;
41  CurriculumVitae = has => ProfessionalExperience;
   CurriculumVitae = has => Education;
43  CurriculumVitae = has => Language;
   CurriculumVitae = has => Technology;
45

   personalinformation = iof => PersonalInformation[name = "John Doe" ,
47  jobDescription = "Student at University of Minho" , age = "" , gender = ""];
   executivesummary = iof => ExecutiveSummary[description = "Currently , I am a
   student at University of Minho."];
   professionalexperience = iof => ProfessionalExperience[startDate = "
   2018-06-01" , endDate = "2019-12-31" , company = "Inditex" , typeOfBusiness = "" ,
   job = "Salesperson" , activities = "I was responsible for selling apparel." ,
   technologies = ""];

```

```

49   education = iof => Education[startDate = "2017-01-01", endDate = "present",
organization = "University of Minho", subjects = "", qualification = "Masters,
Software Engineering"];
language = iof => Language[language = "English", reading = "Native or
Bilingual", writing = "Native or Bilingual", verbal = "Native or Bilingual"];
51   programminglanguage = iof => ProgrammingLanguage[technology = "", level = "",
startDate = "", endDate = "", description = ""];
database = iof => Database[technology = "", level = "", startDate = "",
endDate = "", description = ""];
53   operatingsystem = iof => OperatingSystem[technology = "", level = "",
startDate = "", endDate = "", description = ""];
tool = iof => Tool[technology = "", level = "", startDate = "", endDate = "",
description = ""]
55 }.

```

Listing 2.2: Description of the ontology with instances

## 2.2 SEMANTIC INFORMATION EXTRACTION

Semantic information extraction is the process of extracting information from unstructured textual sources to enable finding entities as well as classifying them. In order to compare the applicants' CVs with the job offers, first, it is necessary to extract the relevant information from them, these being the fields stated in the ontology defined in Subsection 2.1.7. Hence, in this section, a brief explanation about entities and relationships will be presented, the typical workflow of Information Extraction will be displayed, the most common Named Entity Recognition (NER) methodologies will be explained and, lastly, examples of application of each step of the IE workflow will be shown.

### 2.2.1 *Entities and relationships*

Before diving into the Information Extraction workflow, it is important to explain the main type of structures extracted from an unstructured source: entities and relationships.

Entities are typically noun phrases. The most popular form of entities is named entities, like proper names, organizations, locations, *etc.*. Figure 54 shows a text, where "Robert Callahan" was recognized as a person, "Estearn's" and "Texas Air Corp." were recognized as organizations and "Houston" was recognized as a location. (Sarawagi, 2008)

According to Robert Callahan, president of Eastern's flight attendants union, the past practice of Eastern's parent, Houston-based Texas Air Corp., has involved ultimatums to unions to accept the carrier's terms
The unstructured source
According to <b>&lt;Per&gt; Robert Callahan &lt;/Per&gt;</b> , president of <b>&lt;Org&gt; Eastern's &lt;/Org&gt;</b> flight attendants union, the past practice of <b>&lt;Org&gt; Eastern's &lt;/Org&gt;</b> parent, <b>&lt;Loc&gt; Houston &lt;/Loc&gt;</b> -based <b>&lt;Org&gt; Texas Air Corp. &lt;/Org&gt;</b> , has involved ultimatums to unions to accept the carrier's terms
Annotated entities

Figure 54: Entities found in the text

Relationships are defined over two or more related entities. Examples of relationships are "is employee of" between a person and an organization, "location of outbreak" between a disease and a location, among others. Figure 55 shows the same text, where the relationship "Employee\_Of" occurs between a person and an organization and the relationship "Located\_In" occurs between an organization and a location. (Sarawagi, 2008)

According to Robert Callahan, president of Eastern's flight attendants union, the past practice of Eastern's parent, Houston-based Texas Air Corp., has involved ultimatums to unions to accept the carrier's terms
The unstructured source
Robert Callahan <b>&lt;Employee_Of&gt;</b> Eastern's Texas Air Corp. <b>&lt;Located_In&gt;</b> Houston
Relationships

Figure 55: Relationships found in the text

### 2.2.2 Information Extraction workflow

A typical workflow of Information Extraction consists of the following steps:

1. Sentence splitting - The text is split into sentences resorting to sentence boundary indicators (question marks, exclamation points, *etc.*); (Choi et al., 2021)



2. Tokenization - Each sentence is then broken down along a predefined set of delimiters (such as spaces, commas, dots, *etc.*) giving rise to tokens which, in turn, are usually words, digits or punctuation marks; (Sarawagi, 2008)
3. Part-Of-Speech (POS) tagging - After the tokens are generated, a grammatical category from a fixed set, like a noun, a verb, an adjective, an adverb, an article, a conjunct, a pronoun or other parts of speech, is assigned to each token; (Sarawagi, 2008)
4. Lemmatization - The various forms of a token (such as "appeared" or "appears") are mapped to its root form, also known as the lexeme or lemma (*e.g.*, "appear"); (Bird et al., 2009)
5. Named Entity Recognition - Single tokens and/or sets of tokens are recognized as organizations, locations, dates, *etc.*; (Lee et al., 2004)
6. Relationship extraction - Relationships between different entities, such as "is employee of", "is located in", among others, are extracted. (Bird et al., 2009)

Figure 56 illustrates the pipeline and shows how the above mentioned steps interconnect.

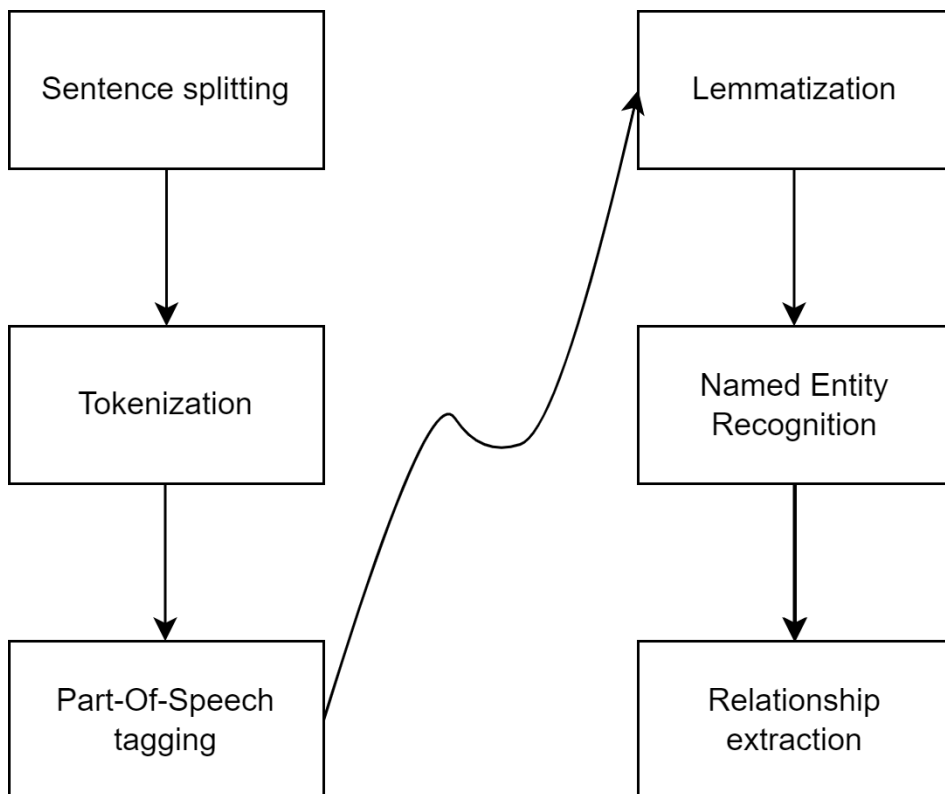


Figure 56: Information Extraction workflow

### 2.2.3 Named Entity Recognition approaches

According to (Choi et al., 2021), Named Entity Recognition approaches are classified into two main types:

- Knowledge Engineering (KE), also referred to as rule-based approach, which uses the domain knowledge of human expertise represented in a machine-understandable form, *i.e.*, in the form of production rules. In this approach, rules are iteratively constructed and refined to improve the accuracy of text processing;
- Automatic training, also known as the Machine Learning approach, which uses ML algorithms, as expected. It requires a large amount of annotated training data.

Comparing the two approaches, it is clear that the efforts required for defining patterns and developing rules in the first approach are less than those required for annotating a large amount of training data in the second approach. Moreover, the first one tends to produce a higher performance, since human expertise results in more accurate patterns. (Choi et al., 2021)

### 2.2.4 Information Extraction workflow with spaCy

spaCy ([spa](#)) is a modern Python library for industrial-strength Natural Language Processing. It comes with pretrained pipelines and currently supports tokenization and training for more than 60 languages. It features state-of-the-art speed and neural network models for tagging, Named Entity Recognition and more.

Therefore, throughout this subsection, the functionalities of spaCy will be explored and applied, performing the steps mentioned in Subsection 2.2.2 in the text sample that follows:

Apple is looking at buying U.K. startup for \$1 billion.

First of all, the library is imported, the English model is loaded which creates an nlp object and the text in question is processed with that nlp object which, in turn, creates a Doc (short for "document") object.

```

1 import spacy
3 nlp = spacy.load("en_core_web_sm")
  doc = nlp("Apple is looking at buying U.K. startup for $1 billion.")

```

Listing 2.3: Import library, load English model and create Doc object

Then, the text is split into sentences.

```

for sent in doc.sents:
    print(sent.text)

```

Listing 2.4: Sentence splitting

In this case, since the text itself consists of a single sentence, the code above results in just that sentence.

Apple is looking at buying U.K. startup for \$1 billion.

After that, that sentence is broken into tokens.

```

for token in doc:
    print(token.text)

```

Listing 2.5: Tokenization

The result contains twelve tokens, including words, symbols, numbers and punctuation marks.

```

Apple
is
looking
at
buying
U.K.
startup
for
$
1
billion
.

```

Next, comes the POS tagging phase where a grammatical category from a fixed set is assigned to each token.

```

for token in doc:
    print(token.text, token.tag_)

```

Listing 2.6: POS tagging

The result shows the detailed POS tag assigned to each token, including "NNP" (noun, proper singular), "VBZ" (verb, 3rd person singular present), "VBG" (verb, gerund or present

participle), "IN" (conjunction, subordinating or preposition), "NN" (noun, singular or mass), "\$" (symbol, currency), "CD" (cardinal number) and "." (punctuation mark, sentence closer). (Pen)

```
Apple NNP
is VBZ
looking VBG
at IN
buying VBG
U.K. NNP
startup NN
for IN
$ $
1 CD
billion CD
. .
```

Then, comes the lemmatization phase where the root form of each token in inflectional or derivational form is derived.

```
for token in doc:
    print(token.lemma_)
```

Listing 2.7: Lemmatization

From the result, it can be seen that "is" was derived as "be", "looking" was derived as "look" and "buying" was derived as "buy".

```
Apple
be
look
at
buy
U.K.
startup
for
$
1
billion
.
```

Later, comes the Named Entity Recognition step where, as the name implies, the tokens are recognized as named entities.

```

for ent in doc.ents:
    print(ent.text, ent.label_)

```

Listing 2.8: Named Entity Recognition

The result shows that spaCy found three entities: "Apple" which was recognized as an organization, "U.K." which was recognized as a geopolitical entity and "\$1 billion" which was recognized as money.

```

Apple ORG
U.K. GPE
$1 billion MONEY

```


spaCy also allows visualizing the text sample and its named entities with different colours through displaCy.



Apple **ORG** is looking at buying U.K. **GPE** startup for \$1 billion **MONEY** .

Figure 57: Text sample and its named entities

When applied to a much more complex text sample, it is possible to see the great power of spaCy. It not only identifies the previous entities, but also recognizes "Steve Jobs", "Steve Wozniak" and "Ronald Wayne" as people, "weekly" as a date, "University of Minho" as an organization and "Portugal" as a geopolitical entity.



Apple **ORG** (founded by Steve Jobs **PERSON** , Steve Wozniak **PERSON** and Ronald Wayne **PERSON** ) is looking at buying U.K. **GPE** startup for \$1 billion **MONEY** . I read this once in the weekly **DATE** newspaper of University of Minho **ORG** which, in turn, is located in Portugal **GPE** .

Figure 58: More complex text sample and its named entities

In addition to the features mentioned so far, spaCy has many more. One of the powerful functionalities that might be useful in the future is the dependency parsing. In this step, the dependencies between the tokens (head token and child token) are identified.

```

for token in doc:
    print(token.text, token.dep_, token.head.text, token.pos_,
          [child for child in token.children])

```

Listing 2.9: Dependency parsing

By the result, it is possible to conclude that "Apple" is a nominal subject with "looking" being its head token; "is" is an auxiliary verb with "looking" also being its head token; "looking" is the root of the sentence, which means that it acts as the head of many tokens ("Apple", "is", "at" and "."), but is not a child of any other token; "at" is a preposition with, once again, "looking" being its head token and "buying" being its child token; "buying" is a complement of preposition with "at" being its head token and "startup" and "for" being its child tokens; "U.K." is a compound noun with "startup" being its head token; "startup" is a direct object with "buying" being its head token and "U.K." being its child token; "for" is a preposition with "buying" also being its head token and "billion" being its child token; "\$" is a quantifier with "billion" being its head token; "1" is a compound noun with "billion" also being its head token; "billion" is an object of preposition with "for" being its head token and "\$" and "1" being its child tokens; "." is a piece of punctuation with "looking" being its head token.

```
Apple nsubj looking PROPN []
is aux looking AUX []
looking ROOT looking VERB [Apple, is, at, .]
at prep looking ADP [buying]
buying pcomp at VERB [startup, for]
U.K. compound startup PROPN []
startup dobj buying NOUN [U.K.]
for prep buying ADP [billion]
$ quantmod billion SYM []
1 compound billion NUM []
billion pobj for NUM [$, 1]
. punct looking PUNCT []
```

Like the named entities, spaCy also allows visualizing the text sample (including the tokens' simple Universal Part-Of-Speech tag set (UPOS) (UPO) POS tags) and the dependency parse tree with directed, labeled arcs from head tokens to child tokens through displaCy.

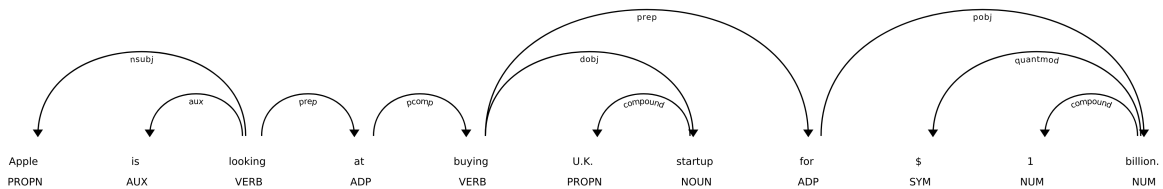


Figure 59: Text sample and its dependency parse tree

## 2.3 SIMILARITY ALGORITHMS

The last step of this Master’s Work is comparing the candidates’ CVs with the job offers. Therefore, in this section of the state of the art, the most popular similarity algorithms (Jaccard Similarity, Term Frequency — Inverse Document Frequency (TF-IDF), Doc2Vec, Bidirectional Encoder Representations from Transformers (BERT) and Universal Sentence Encoder (USE)) will be explored and analyzed.

### 2.3.1 Jaccard Similarity

Jaccard Similarity is defined as the size of the intersection divided by the size of the union of the sample sets, which can be represented by the following formula: (Bank and Cole, 2008)

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \quad (1)$$

Adapting this definition to the problem in question, we can translate Jaccard Similarity as the ratio between the cardinality of words in common in both documents and the number of words resulting from the union of both documents. Follows an illustrative example to aid understanding.

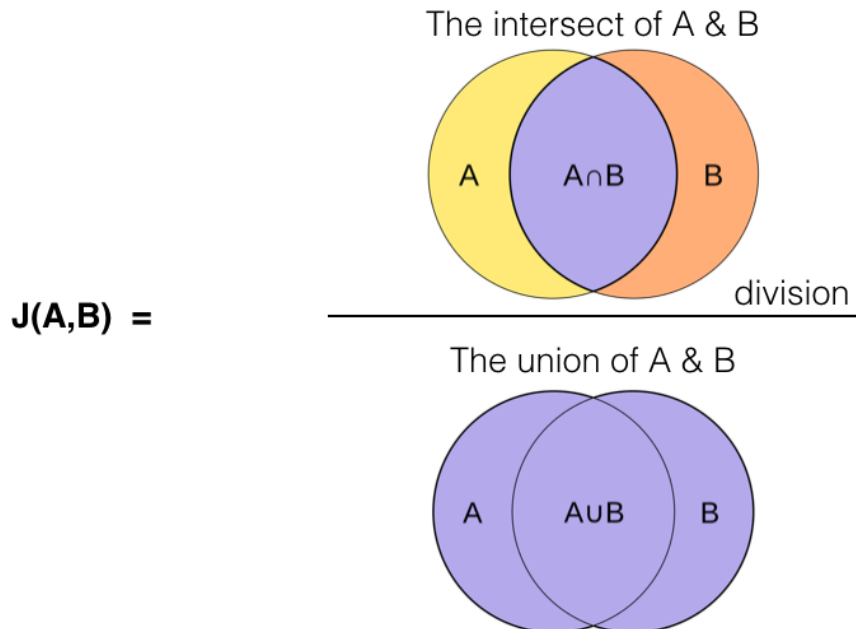


Figure 60: Jaccard Similarity

### 2.3.2 TF-IDF

TF-IDF measures how much a term (word) is relevant to a document (set of words) in a collection of documents, which can be represented by the following formula:

$$TFIDF(t, d) = TF(t, d) \times IDF(t) \quad (2)$$

Term Frequency  $TF(t, d)$  measures the number of times term  $t$  occurs in document  $d$ . (Jing et al., 2002) However, this depends on the generality of the term and the length of the document. For example, the term "is" will probably appear more times in a 10000 worded document than in a 100 worded one, but it is not correct to say that the first document is more important. Therefore, the absolute frequency is divided by the total number of terms in the document.

$$TF(t, d) = \frac{\text{count of } t \text{ in } d}{\text{number of terms in } d} \quad (3)$$

Document Frequency  $DF(t)$  is the number of documents in which the term  $t$  occurs at least once. (Jing et al., 2002)

$$DF(t) = \text{occurrence of } t \text{ in document set} \quad (4)$$

Similarly to Term Frequency, Document Frequency is divided by the total number of documents  $N$ .

The main goal is to know the informativeness of a term, which means that, ideally, the more common a term is (e.g., stop words, such as "the", "a", "an", "so", "what", etc.), the lower the weight. However,  $\frac{DF(t)}{N}$  returns precisely the inverse and, so, it is necessary to invert this fraction. Still, a large  $N$  will result in a huge value for the latter, hence it is essential to take the log of it which, in turn, originates Inverse Document Frequency  $IDF(t)$ .

$$IDF(t) = \log\left(\frac{N}{DF(t)}\right) \quad (5)$$

### 2.3.3 Doc2Vec

Doc2Vec, as the name implies, is capable of transforming text documents into a vectorized form and is heavily based on Word2Vec, as expected. Therefore, first, it is important to briefly explain the latter.

Like with text documents, words also need to be converted into vectors, so that machines can understand them. A traditional approach is simply one-hot encode the words, which results in vectors (whose length is equal to the size of the total unique vocabulary in the



corpus) with only one target element being 1 and the remaining being 0. Conventionally, these words are encoded in alphabetical order, which means that one-hot vectors for words starting with "a" will have value 1 at lower indexes while those for words beginning with "z" will have the same value at higher indexes. However, this approach has, at least, two issues. First, it is not possible to infer the relationship between two words. For instance, although they have a similar meaning, "endure" and "tolerate" will have their targets far away from each other. Additionally, there are a numerous redundant 0 values in the vectors which, in turn, are wasting a lot of space.

"a"	"abbreviations"	"zoology"	"zoom"
1	0	0	0
0	1	0	1
0	0	0	0
⋮	⋮	⋮	⋮
0	0	0	0
0	0	1	0
0	0	0	1

Figure 61: One-hot encoding

Word2Vec is an efficient solution to these problems, because it is a Word Embedding method and, so, generates a type of mapping that allows words with similar meaning to have similar representation. It can be divided into two subtypes: skip-gram and Continuous Bag Of Words (CBOW).

For skip-gram, the input is the target word, while the outputs are the words surrounding it. For instance, in the sentence "I have a cute cat", if the input was "a", the corresponding output would be "I", "have", "cute" and "cat", assuming the window size is 2.

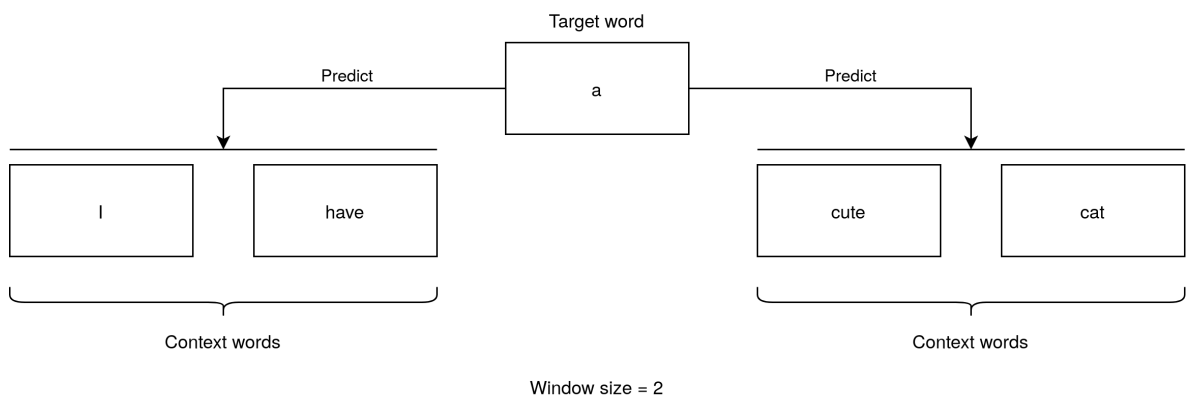


Figure 62: Skip-gram

CBOW is very similar to skip-gram, except that it swaps the input and outputs, *i.e.*, given a context, CBOW predicts which word is most likely to appear in it.

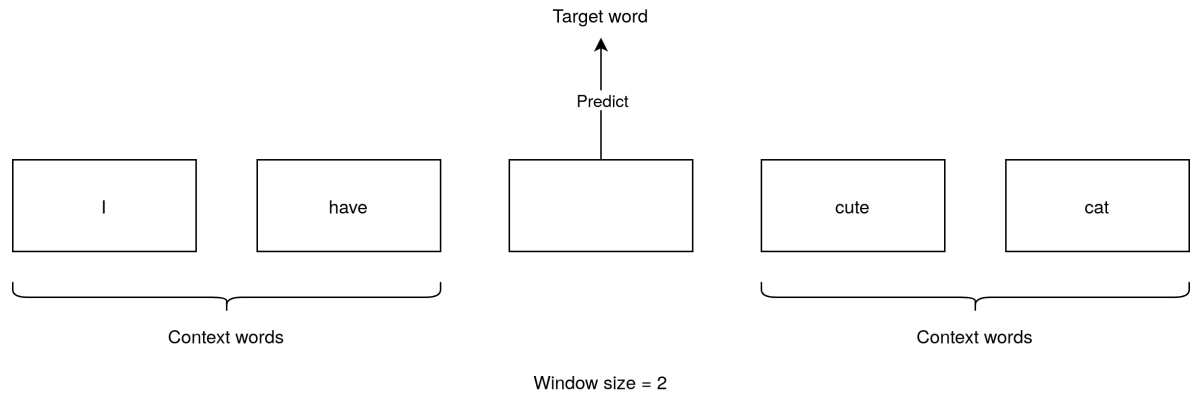


Figure 63: CBOW

After understanding Word2Vec, it is easier to understand how Doc2Vec works. As mentioned before, the goal of Doc2Vec is to create a vector representation of a document. But unlike words, documents do not come in logical structures, so two variants of skip-gram and CBOW, respectively, were proposed by (Le and Mikolov, 2014).

The first variant, Distributed Bag Of Words version of Paragraph Vector (PV-DBOW), instead of using the target word to predict the context words, it uses the paragraph unique identifier.

The second variant, Distributed Memory version of Paragraph Vector (PV-DM), in addition to using the context words to predict the target word, it uses the paragraph unique identifier.

#### 2.3.4 BERT

As mentioned before, BERT stands for Bidirectional Encoder Representations from Transformers, so it is important to start by explaining what Transformer is.

Transformer is a deep learning model designed to handle sequential data, such as natural language, proposed in (Vaswani et al., 2017). However, it does not require the sequential data to be processed in order, *i.e.*, if the input data is a natural language sentence, Transformer does not need to process the beginning of it before the end.

Transformer is based on an encoder-decoder architecture that, as expected, comprises encoders - which consist of a set of encoding layers that process the input iteratively one layer after another - and decoders - that consist of a set of decoding layers that do the same thing to the output of the encoder. Thus, when a sentence is passed into Transformer, it is embedded (mapped to a vector) and transferred into a stack of encoders. The output from

the final encoder is then passed into each decoder block in the decoder stack. The decoder stack then generates the output.

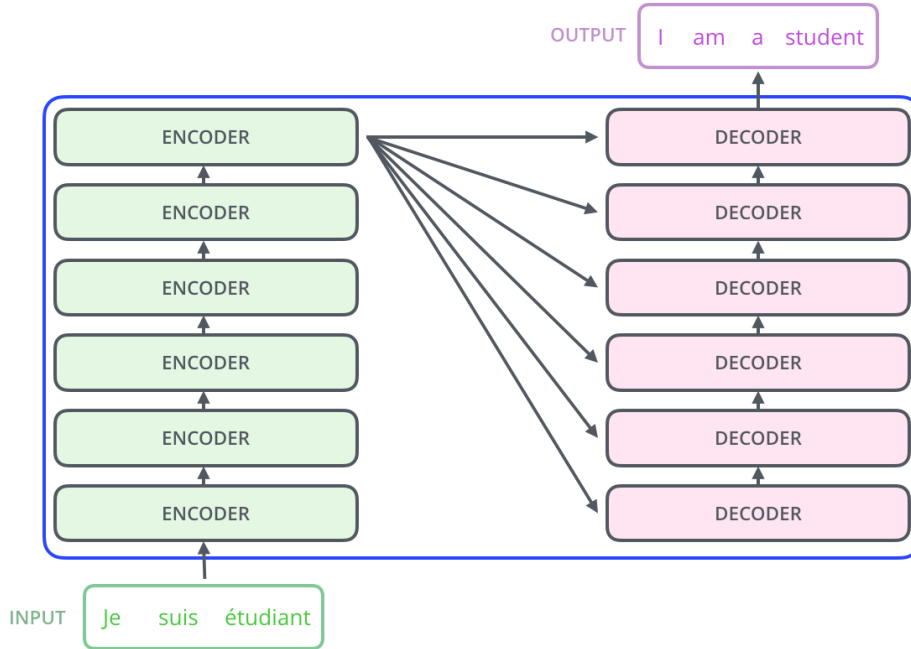


Figure 64: Transformer’s architecture

After describing Transformer’s architecture, it is now easier to explain BERT’s one. Inside Transformer, the encoder cells are used to read the input sentence and the decoder cells are used to predict the output sentence. However, in the case of BERT, since the goal is a model that reads all the words in the input sentence simultaneously (hence the bidirectional) and generates some contextualized embeddings that can be used for various NLP tasks, only the encoder part of Transformer is used.

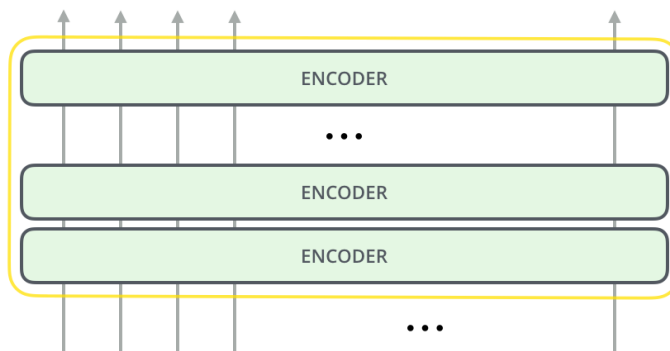


Figure 65: BERT’s architecture

## 2.3.5 USE

The Universal Sentence Encoder encodes text into high dimensional vectors that can be used for numerous natural language tasks.

(Cer et al., 2018) describes two main architecture types: one based on Transformer's architecture and another based on Deep Averaging Network (DAN).

The first one builds embeddings using the encoder module of the Transformer architecture which, as mentioned before, computes a context aware representation of words in the input sentence.

The second one computes the uni-gram and bi-gram embeddings first and then averages them to get a single embedding. This is, then, passed to a Deep Neural Network (DNN) to get a final sentence embedding.



Figure 66: USE's input and output paradigm

## 2.3.6 Algorithm comparison

Algorithm	Pros	Cons
Jaccard Similarity	It is good for cases where duplication does not matter.	It is highly influenced by the size of the documents. Large documents can have a big impact on the similarity as it could significantly increase the union whilst keeping the intersection similar.
TF-IDF	TF-IDF is simple to calculate.	TF-IDF will suddenly plummet to zero if the word of interest appears in all of the documents, even if it only appears once in each of them.
Doc2Vec	It takes word order into account.	Accuracy depends on document length.
BERT	Much better model performance over legacy methods.	It requires huge computational resources.
USE	No preprocessing is needed.	USE supports only 16 languages.

Table 1: Comparison between the algorithms

Through the description of each algorithm and the analysis of this table, it is clear that each of them has its pros and cons, but BERT stands out from the rest as it presents state-of-the-art results in a wide variety of NLP tasks and, therefore, has been widely used since its publication.

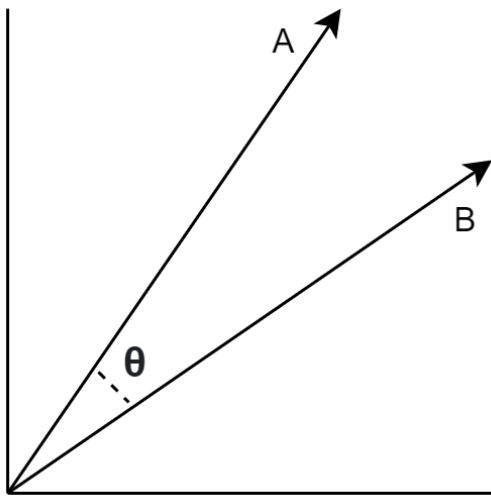
To reinforce this idea, some papers linked to Similarity Relatedness (which, in turn, corresponds to the degree of semantic relatedness between a pair of sentences (Fonseca and Alvarenga, 2020)) were studied and the conclusion was the same. In fact, (Rodrigues et al., 2019) claim that they resorted to BERT algorithm; (Santos et al., 2019) state that their solution is worse than solutions based on Deep Learning, including Transformer based algorithms, such as BERT; (Fonseca and Alvarenga, 2020) also declare that their BERT based approach is the best one and (Souza et al., 2019) are willing to take advantage of contextual Word Embeddings, such as those resulting from the application of BERT algorithm. In other words, they all apply BERT algorithm and/or imply that its application is advantageous.

## 2.4 SIMILARITY MEASURES

Since most of the explored algorithms are vectorization algorithms (*i.e.*, algorithms that turn documents into word embeddings), there is a need to explore some similarity measures, in order to compute the affinity between the vectors corresponding to the CVs and to the job offers. Therefore, in this last section of the state of the art, some of the most popular similarity measures in NLP (cosine similarity, Euclidean distance, Manhattan distance and Chebyshev distance) will be studied.

### 2.4.1 Cosine similarity

The cosine similarity measures the similarity between two vectors of an inner product space (Han et al., 2011). It is measured by the cosine of the angle between those vectors (Han et al., 2011), that is, the dot product of the vectors divided by the product of their lengths ( $\cos$ ) (as can be seen in Figure 67). The cosine similarity always belongs to the interval  $[-1,1]$ . For example, two proportional vectors have a cosine similarity of 1, two orthogonal vectors have a similarity of 0 and two opposite vectors have a similarity of -1 ( $\cos$ ).



$$\cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}$$

Figure 67: Cosine similarity

#### 2.4.2 Euclidean distance

The Euclidean distance finds the distance between any two points in Euclidean space which are also called Euclidean vectors (**vec**). It is calculated as the square root of the sum of the squared differences between the two vectors (**mea**), which can be represented by the following formula:

$$d(u, v) = \sqrt{\sum_{i=1}^n (u_i - v_i)^2} \quad (6)$$

#### 2.4.3 Manhattan distance

The Manhattan distance calculates the distance between two real-valued vectors (**mea**). It is computed as the sum of the absolute differences between the two vectors (**mea**), which can be represented by the following formula:

$$d(u, v) = \sum_{i=1}^n |u_i - v_i| \quad (7)$$

#### 2.4.4 Chebyshev distance

The Chebyshev distance determines the distance between two vectors on a vector space (che). It is calculated as the greatest of their differences along any coordinate dimension (che), which can be represented by the following formula:

$$d(u, v) = \max_i (|u_i - v_i|) \quad (8)$$

## 2.5 SUMMARY

In this second chapter, the state of the art regarding the scope of this Master's Work was analyzed. In particular, regarding CV sources and formats, the Europass, LinkedIn profile as a CV, Curriculum Lattes, CIÊNCIAVITAE and Syone's internal CV were investigated and the ontology that decides the information to be extracted from each CV was defined. With respect to semantic information extraction, the concepts of entities and relationships and the NER approaches were explained along with the presentation of the typical IE workflow. Lastly, the most popular similarity algorithms (Jaccard Similarity, TF-IDF, Doc2Vec, BERT and USE) and the most popular similarity measures (cosine similarity, Euclidean distance, Manhattan distance and Chebyshev distance) were explored. Thus, let us move on to the presentation of the proposed architecture for the system.

---

## PROPOSED APPROACH

---

The objective of this Master's Project is to develop an algorithm capable of identifying the most suitable candidates for a given job offer. Therefore, in this chapter, an architecture proposal for the system was designed and explained, in order to facilitate the understanding of the entire project.

### 3.1 SYSTEM ARCHITECTURE

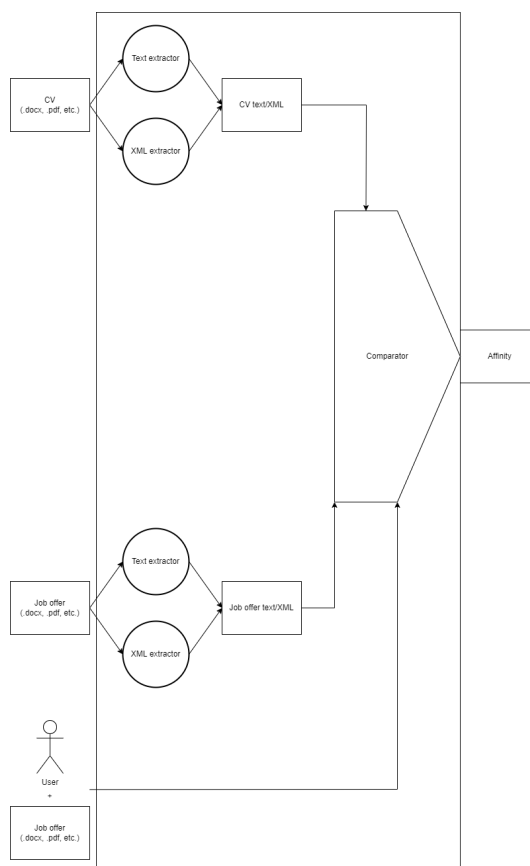


Figure 68: General architecture of the system



Figure 68 depicts the general architecture of the proposed software system to be implemented in this project, in order to accomplish the objectives defined in Chapter 1. The diagram shown sketches the main blocks of the system and its overall characteristics, without considering all details or specific aspects that will be further refined. In this architecture, it is shown that the system will receive a CV (provided in .docx, .pdf, etc. format) which, in turn, will go through the text extractor engine and/or the XML extractor engine, to extract its text and/or XML, respectively, and be properly processed in the next phase.

In parallel, the system will also receive a job offer that can be made available in two ways. It will either receive a file (also provided in .docx, .pdf, etc. format) which, in this case, will undergo an analogous preprocessing or the user will introduce the job offer’s relevant information directly.

At this point, the CV’s information plus the job offer’s information will enter a comparator module that, as the name suggests, compares the CV’s data and the job offer’s requirements and returns the affinity between them, quantifying how much the individual’s details comply with the company’s requests.

Keep in mind that, to develop the comparator module, two different approaches will be tried, aiming at a clear understanding of the pros and cons of each one of them.

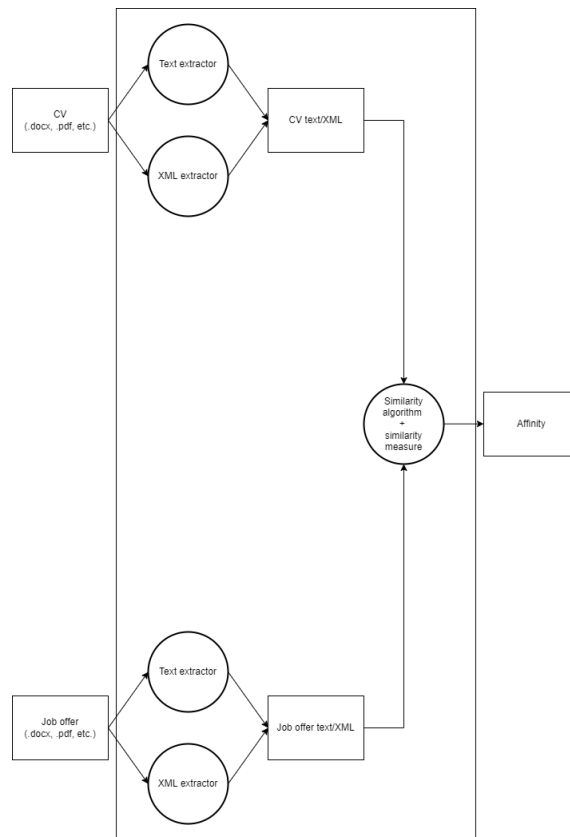


Figure 69: Architecture of approach A

The architecture of the first approach can be seen in Figure 69. In this architecture, it is shown that, with regard to the job offer, the first path mentioned in the general architecture will be followed. Therefore, both the CV text and/or XML and the job offer text and/or XML will go through an engine that will run a Machine Learning-inspired similarity algorithm and a similarity measure on these inputs and return the affinity between them.

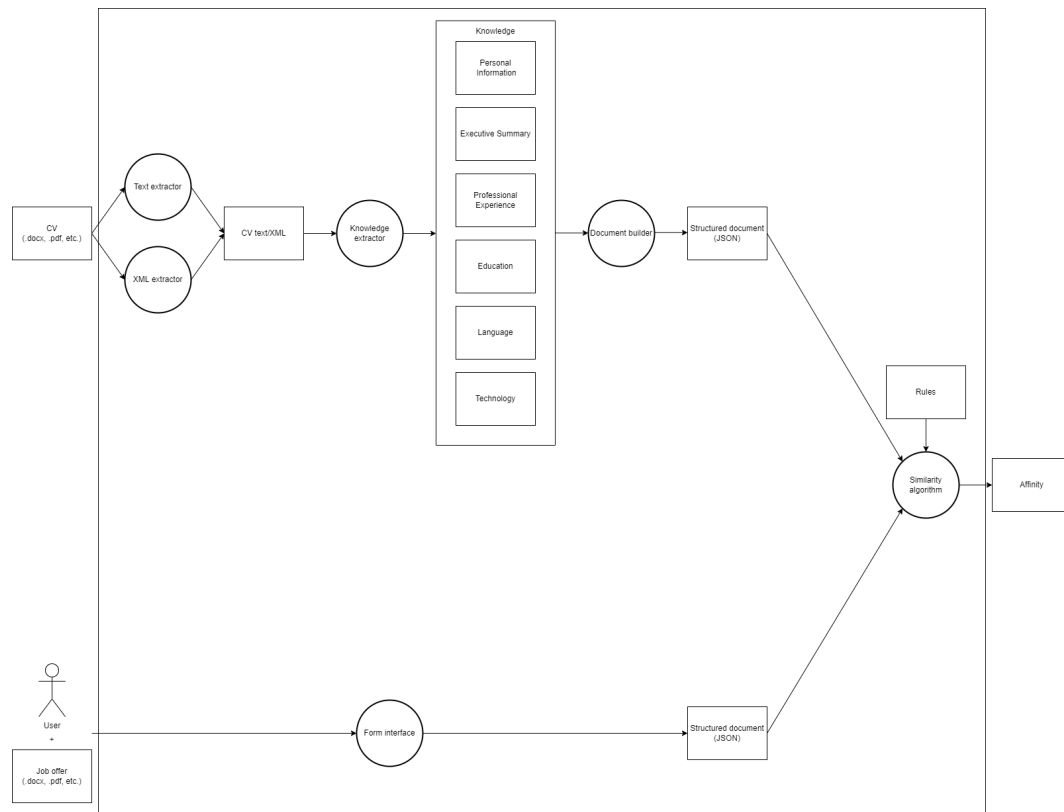


Figure 70: Architecture of approach B

The architecture of the second approach can be seen in Figure 70. In this architecture, it is shown that the knowledge, that is, all the relevant information from the CV text and/or XML (*i.e.*, the attributes associated with the concepts defined in the ontology presented in Subsection 2.1.7 (Personal Information, Executive Summary, Professional Experience, Education, Language and Technology)) needs to be extracted, resorting to the knowledge extractor engine. After the data extraction process, the collected data is gathered and properly structured in an internal representation that corresponds to the content of the CV and, from there, the document builder engine creates a structured document (in this case, a JavaScript Object Notation (JSON) document).

In parallel, in this approach, regarding the job offer, the second path referred to in the general architecture will be followed, *i.e.*, the user will fill in a form with the relevant information about the job offer in question (such as the area and education level requested)

and, the associated interface module, in turn, will return the corresponding structured document (in this case, a JSON document).

Lastly, the structured documents together with the rules that define the weights of each requirement will be passed through the similarity algorithm engine which will return the final output, *i.e.*, the affinity between the CV and the job offer.

The algorithms designed and implemented to achieve the first approach and the second approach will be described and discussed in detail in Chapter 5 and Chapter 6, respectively.

### 3.2 SUMMARY

In this chapter, it was presented the proposed approach to solve the main challenge. This approach, in turn, was subdivided into two other approaches that will be tried in order to assess the pros and cons of each one of them. To sum up, in order to find the affinity between the CV and the job offer, the first approach will run a ML algorithm and a similarity measure on the inputs while the second approach will compare the inputs by structuring them first. With that in mind, let us start by explaining the approaches' engines involved in processing CVs.

---

## CVS PROCESSING

---

This chapter aims to clarify not only the work carried out in terms of CVs processing, but also to describe the challenges that have arisen along this path. Therefore, in the next sections, a detailed explanation of the text extractor, XML extractor, knowledge extractor and document builder engines mentioned in Chapter 3 will be given. Note that the knowledge extractor and the document builder engines are only used in the comparator module of the second approach. It should also be noted that, from this chapter onwards, only Syone's internal CV format will be considered, but, in the future, other CV formats may be dealt with following an analogous process.

### 4.1 TEXT EXTRACTOR AND XML EXTRACTOR

As shown in the general architecture of the system, it is necessary to extract the text and/or XML of the document itself.

For that, the Extractor class was developed aided by two subclasses: TextExtractor and XMLExtractor. As their names imply, the TextExtractor subclass is used to extract the content of documents in plain text format while the XMLExtractor subclass is used to extract XML from documents, allowing a more structured analysis. Note that the first one takes advantage of the Apache Tika library through the bindings provided by a Python package (`tik`) which, in turn, is capable of handling different file types, such as `.pdf`, `.docx`, *etc.*. Also note that, regarding the second one, the logic involves accessing the XML from the `.docx` files (which are actually `.zip` files with all the XML files associated with the document in question), as the documents to be processed are in this format. If these were in a different file format (like `.pdf`), the methodology for extracting their XML would be completely different.

The class diagram of the text extractor and XML extractor engines can be seen in Figure 71.

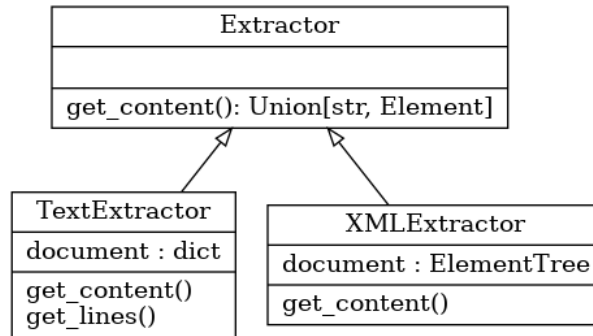


Figure 71: Text extractor and XML extractor engines' class diagram

## 4.2 KNOWLEDGE EXTRACTOR

After extracting the text and/or XML from each CV, it is now possible to extract its sections (Personal Information, Executive Summary, Professional Experience, *etc.*), to later build the corresponding structured document.

Therefore, first, the Parser class was created followed by one subclass: CVSyoneParser (capable of extracting information from CVs following Syone's template). Note that, with this architecture, in the future, it will be possible to add more subclasses capable of parsing different CV formats (*e.g.*, EuropassParser specialized in extracting information from Europass CVs, LinkedInParser specialized in extracting information from LinkedIn profiles as CVs, *etc.*).

Regarding CVSyoneParser, the reasoning for extracting text from the Personal Information, Executive Summary, Professional Experience, Education and Languages sections turned out to be always the same: find the word that indicates the beginning of the section and the word that indicates the end of it (which also indicates the beginning of the next section) and extract the text between them. To ensure that the captured terms are effectively headings and not random words throughout the document, these words are searched along with a newline character. This is done with resource to the `re` module (`re`) which provides many functions to deal with regular expressions.

Let us take a look at an example. Figure 72 shows an example of an Education section which starts with the keyword "EDUCATION" and ends with the keyword "LANGUAGES" (which, in turn, symbolizes the beginning of another section). From here, it is possible to conclude that, in order to extract the Education section's text, it is necessary to extract the text between those two keywords.

**EDUCATION**

Dates (from to)	September 2001 to July 2007
Name and type of organization providing education and training	Faculdade de Ciências da Universidade do Porto
Principal subjects / occupational skills covered	<ul style="list-style-type: none"> <li>• Networks and IT Systems administration and security</li> <li>• Network architecture</li> <li>• Network applications development and deployment</li> <li>• Cryptography</li> <li>• UNIX and Windows operating systems</li> <li>• Programming skills in: C, Java, J2EE, Perl, Python, Bash Script, Haskell, Prolog, and LATEX</li> <li>• Web: PHP, HTML, CSS, JavaScript</li> <li>• Data Bases: MySQL and PostgreSQL</li> <li>• Advanced network topics such as SIP, SNMP, MPLS, Multicast, MIP, WEP, WPA</li> <li>• Software Engineering (and software design patterns).</li> </ul>
Title of qualification awarded	Bachelor's Degree in Networks and IT Systems Engineering

**LANGUAGES**

Other Languages	Reading Skills	Writing Skills	Verbal Skills
-----------------	----------------	----------------	---------------

Figure 72: Example of an Education section

When it came to extracting the fields from the Personal Information section (in this case, name, job description, age and gender) the logic was similar.

The process started to get more complex in the sections that contained more than one entry (Professional Experience and Education). Initially, an approach was followed where the indexes that mark the beginning and the end of all keywords (Dates (from to), Name and type of organization offering education and training, *etc.*) were stored in data structures (in this case, lists). Then, according to the length of those lists, in each iteration, the text of the different fields was extracted between the indices representing the respective delimiters. However, this method proved to be impractical because, when some fields were missing, the length of all data structures was not the same. Therefore, there was a need to embark on

another approach. This time, those entries were divided by the separator "Dates (from to)" (see how each entry starts with those keywords on Figure 73 and Figure 74).

### PROFESSIONAL EXPERIENCE

Dates (from to)	26/01/2021 – Current
Company Name	BNP Paribas (Syone)
Type of business or sector	Financial Services
Occupation or position held	Technical Support
Main activities and responsibilities	<p>During this experience, the consultant was responsible for the following activities:</p> <ul style="list-style-type: none"> <li>- Checking the application is executing the flow accordingly.</li> <li>- Crossing data in the application with the database via TSO and delivering extraction regarding some topic via TSO.</li> <li>- Investigate issues where the application could be involved, meaning trying to understand the flow for that issue and understand with which other application is the issue related and with the team responsible come up with a fix.</li> </ul>
Software and Environments used	<ul style="list-style-type: none"> <li>- TSO</li> <li>- SQL</li> <li>- 2Strack</li> <li>- Service-Now</li> </ul>
Dates (from to)	19/04/2020 – 23/01/2021

Figure 73: Separator of each Professional Experience section's entry

### EDUCATION

Dates (from to)	September 2019 – January 2022
Name and type of organization providing education and training	University of Minho, Braga
Principal subjects / occupational skills covered	<ul style="list-style-type: none"> <li>• Big Data Processing using Apache tools such as Hadoop and Spark;</li> <li>• Machine Learning problem solving tools such as Tensorflow and Keras, using all types of known techniques;</li> <li>• Architecture building tools for constructing an intelligent system.</li> </ul>
Title of qualification awarded	Master's in Data Science and Intelligent Systems
Dates (from to)	September 2016 – June 2019

Figure 74: Separator of each Education section's entry

Then, the main reasoning for extracting the fields from each entry was similar to the Personal Information section, with special attention to some cases, such as:

- Professional Experience entries without the "Software and Environments used" field;

Dates (from to)	Apr 2016 – Jun 2019
Company Name	Exército Português
Type of business or sector	Ministry of Defense
Occupation or position held	Corporal (OR-3)
Main activities and responsibilities	Responsible for a team of 6 persons

Figure 75: Example of a Professional Experience entry without the "Software and Environments used" field

- Professional Experience entries without the "Occupation or position held" field;

Dates (from to)	2015-2018
Company Name	Freelance (Fleetwood Consulting, ALM Pilates)
Type of business or sector	IT
Main activities and responsibilities	<p>During this experience, the consultant was responsible for the following activities:</p> <ul style="list-style-type: none"> <li>• Advising on Software and Hardware acquisitions.</li> <li>• Configuring automated cloud backups of documents and sensitive information.</li> <li>• Configuration of remote access and management.</li> <li>• Setting up sales front, as well as configuration of billing software.</li> </ul>

Figure 76: Example of a Professional Experience entry without the "Occupation or position held" field

- Education entries without the "Principal subjects / occupational skills covered" field;



Dates (from to)	September 2018 – June 2021
Name and type of organization providing education and training	Instituto Superior Técnico
Title of qualification awarded	Master’s in computer engineering

Figure 77: Example of an Education entry without the "Principal subjects / occupational skills covered" field

- Education entries with multiple "Title of qualification awarded" fields;

Dates (from to)	2016-2019
Name and type of organization providing education and training	Instituto Superior de Engenharia do Porto (ISEP)
Title of qualification awarded	Licenciatura em Engenharia Biomédica
Title of qualification awarded	...

Figure 78: Example of an Education entry with multiple "Title of qualification awarded" fields

- Education entries without the "Title of qualification awarded" field;

Dates (from to)	2014
Name and type of organization providing education and training	IEFP
Principal subjects / occupational skills covered	<p>Modular training by the IEFP in partnership with everis in the following short-term training units of the National Qualification Catalogue:</p> <ul style="list-style-type: none"> <li>• 0812 – SQL</li> <li>• 0814 – SQL – advanced</li> <li>• 0807 – COBOL – first principles</li> <li>• 0808 – COBOL – files and interactivity</li> </ul>

Figure 79: Example of an Education entry without the "Title of qualification awarded" field

Like the Professional Experience and Education sections, the Languages table also has several entries (in this case, rows), so, before extracting its fields (Other Languages, Reading Skills, Writing Skills and Verbal Skills), it was necessary to obtain those rows. After consulting several CVs, it was found that each row was separated by two newline characters, so the table text was split on that separator to get them. Finally, to acquire the fields, the rows were split on a newline character because, once again, this pattern was found to be common to all CVs.

As for the sections referring to the Technologies table (Programming Languages, Databases, Operating Systems and Tools) the reasoning turned out to be a little more complex than manipulating the text extracted by the TextExtractor. In the early stages of development this was attempted because, like the Languages table, each row appeared to be separated by two newline characters and each field (Technology, Level, Duration, Date and Detail Description) appeared to be separated by one newline character. However, it was not that simple. Many CVs had more than one newline character separating each field or even a newline character in the middle of each field's text, which made this approach impossible. So, instead, there was a need to use the XMLExtractor to extract the XML corresponding to the table and, consequently, do a much more structured manipulation.

In addition to the obstacles that have been exposed so far, there was another challenge (common to several sections) related to dates. Take into account that this challenge was solved with the help of the re module.

First, it was necessary to find the temporary start date and temporary end date for each date field. However, many problems arose at this stage alone, such as candidates with invalid date formats, candidates with date formats from which it was not possible to know

whether it was a start date or an end date, among others. Note that these cases were handled by assigning an empty string to both the final start date and the final end date.

Dates (from to)	1 week 2018
Company Name	Critical Software
Type of business or sector	Software Development
Occupation or position held	Summer Innovation Week Student

Figure 80: Candidate with an invalid date format

Dates (from to)	15/11/2021
Name and type of organization providing education and training	Microsoft Certified Azure Fundamentals
Principal subjects / occupational skills covered	<ul style="list-style-type: none"> <li>• Describe cloud concepts (20-25%)</li> <li>• Describe core Azure services (15-20%)</li> <li>• Describe core solutions and management tools on Azure (10-15%)</li> <li>• Describe general security and network security features (10-15%)</li> <li>• Describe identity, governance, privacy, and compliance features (15-20%)</li> <li>• Describe Azure cost management and Service Level Agreements (10-15%)</li> </ul>
Title of qualification awarded	Microsoft Certified Azure Fundamentals – AZ-500

Figure 81: Candidate with a date format from which it was not possible to know whether it was a start date or an end date

Still at this stage, another problem involved candidates not using the same separator to break apart the start date from the end date (some used "to", others used "till", *etc.*). The solution to this problem consisted in transforming all these different separators into a single one in common: the hyphen.

Dates (from to)	January 2019 to January 2021
Company Name	Adentis (Client: Eximbills Technologies)
Type of business or sector	Banktrade
Occupation or position held	Junior Developer

Figure 82: Candidate using "to" to separate the start date from the end date

Dates (from to)	2021 till date
Company Name	Syone
Type of business or sector	Professional Services
Occupation or position held	Associate Engineer

Figure 83: Candidate using "till" to separate the start date from the end date

After this, if the date in question consisted of four digits plus two digits plus two more digits divided by a separator like a hyphen and followed by any single character (*e.g.*, 2021-11-02 - now), it was split on the hyphen surrounded by white spaces so that the temporary start date consisted of everything before (*e.g.*, 2021-11-02) and the temporary end date consisted of everything after (*e.g.*, now).

If the date in question consisted of two digits followed by characters and two digits followed by characters followed by four digits separated by a hyphen (*e.g.*, 01 october - 15 june 2011), since it clearly represents a start date and an end date during the same year, the temporary start date came to consist of everything before the hyphen plus the final four digits (*e.g.*, 2011-10-01) and the temporary end date came to consist of everything after the hyphen (*e.g.*, 2011-06-15). If the date in question did not identify with the previous description (*e.g.*, 2016; 2019-2022), it was split on the hyphen, so that the temporary start date consisted of everything before the first hyphen occurrence (*e.g.*, 2016; 2019) and the temporary end date consisted of everything after the last hyphen occurrence (*e.g.*, 2022). Then, if the temporary start date had a comma or a semicolon (*e.g.*, 2016; 2019), it came to consist of everything before that comma or semicolon (*e.g.*, 2016). On the other hand, if the temporary end date had a comma or a semicolon (*e.g.*, 2017; 2018), it came to consist of everything after that comma or semicolon (*e.g.*, 2018).

If the date in question had the respective start date and end date separated by a comma or a semicolon (*e.g.*, 2017, 2021), those dates were split on one of those separators (depending on the case), so that the temporary start date consisted of everything before the separator (*e.g.*, 2017) and the temporary end date consisted of everything after (*e.g.*, 2021).

If the date in question consisted of two digits plus four digits divided by a slash followed by two digits plus four digits divided by a slash (*e.g.*, 04/2017 07/2017), the temporary start date came to consist of the first group (*e.g.*, 04/2017) and the temporary end date came to consist of the last group (*e.g.*, 07/2017).

If the date in question consisted of four digits followed by four digits (*e.g.*, 2020 2021), the temporary start date came to consist of the first group of four digits (*e.g.*, 2020) and the temporary end date came to consist of the last group of four digits (*e.g.*, 2021).

If the date in question consisted of four digits plus four more digits separated by a slash (*e.g.*, 2020/2021), the temporary start date came to consist of everything before the slash (*e.g.*, 2020) and the temporary end date came to consist of everything after the slash (*e.g.*, 2021).

If the date in question did not identify with any of these cases (*e.g.*, 2018), both the start date and the end date became equal to it.

Once having the temporary start and end dates, it was necessary to deal with each one of them individually. Bear in mind that most cases were handled with the help of the datetime module (`dat`, `b`) and the dateparser library (`dat`, `a`).

Regarding the temporary start date:

- If it only consisted of four digits (*e.g.*, 2021), the final start date came to consist of these four digits as the year, plus January as the month and 1 as the day (*e.g.*, 2021-01-01), since if one only has access to the year, it makes sense that the final start date is the first day of the first month of that year;
- If it consisted of two digits plus four digits divided by a separator like a hyphen (*e.g.*, 09-2020), since that usually represents the month and the year, respectively, the final start date came to consist of those digits plus the first day of the month (*e.g.*, 2020-09-01);
- If it consisted of four digits plus two digits plus two more digits divided by a separator like a hyphen (*e.g.*, 2021-10-29), since that usually represents the year, the month and the day, respectively, the final start date remained the same;
- If it consisted of word characters plus two digits separated by a slash (*e.g.*, oct/21), since that usually represents the month and the year, respectively, the final start date came to consist of those word characters and digits plus the first day of the month (*e.g.*, 2021-10-01);

- If it did not identify with any of these cases nor did it contain "ccna v7.0" (since there is a temporary start date equal to that) (*e.g.*, 02/03/2021), the final start date remained pretty much the same, except for a change in the separators and order (*e.g.*, 2021-03-02);

As for the temporary end date:

- If the candidate used words that indicated that the professional experience, education or technology in question had not yet ended, such as "current", "today", "at this moment", *etc.*, the final end date became "present";
- If the temporary end date consisted of four digits (*e.g.*, 2022) and those four digits corresponded to the current year, the final end date also became "present";
- If the temporary end date consisted of the description given in the first case listed for the temporary start date and those four digits were greater than 1900 (since there are temporary end dates like 218 which are clearly typos), the final end date was calculated analogously, except for the month and the day which became December and 31, respectively (*e.g.*, 2021-12-31);
- If the temporary end date consisted of the description given in the second case listed for the temporary start date, the final end date was calculated analogously, except for the day which became the last day of the month in question (*e.g.*, 2020-09-30);
- If the temporary end date consisted of the description given in the third case listed for the temporary start date, the final end date was calculated analogously;
- If the temporary end date consisted of the description given in the fourth case listed for the temporary start date, the final end date was calculated analogously, except for the day which became the last day of the month in question (*e.g.*, 2021-10-31);
- If the temporary end date did not identify with any of these cases and did not consist of five digits (since there are temporary end dates like 20222 which are also typos) nor did it contain "networks" (since there is a temporary end date equal to "introduction to networks") (*e.g.*, 23/01/2021), the final end date remained pretty much the same, except for a change in the separators and order (*e.g.*, 2021-01-23). Note that if this resulting date was greater than or equal to the current date (*e.g.*, 31/12/2022), the final end date became "present".

Finally, the class diagram of the knowledge extractor engine can be seen in Figure 84, Figure 85 and Figure 86.

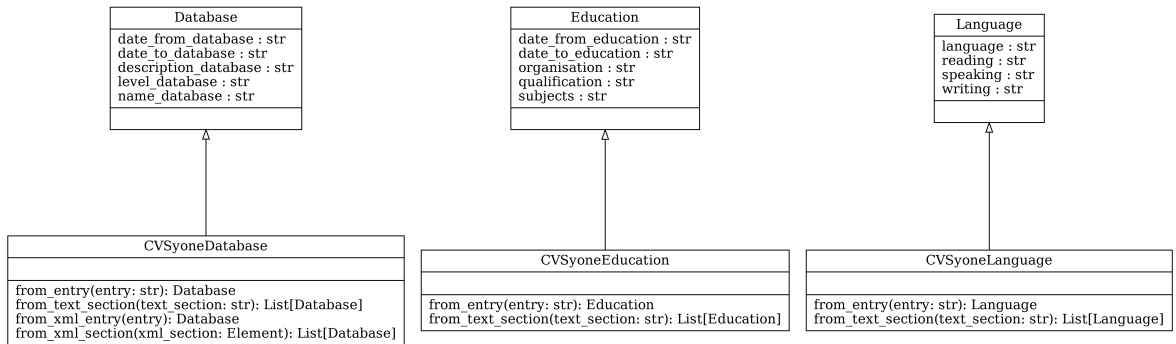


Figure 84: Knowledge extractor engine’s class diagram (part 1)

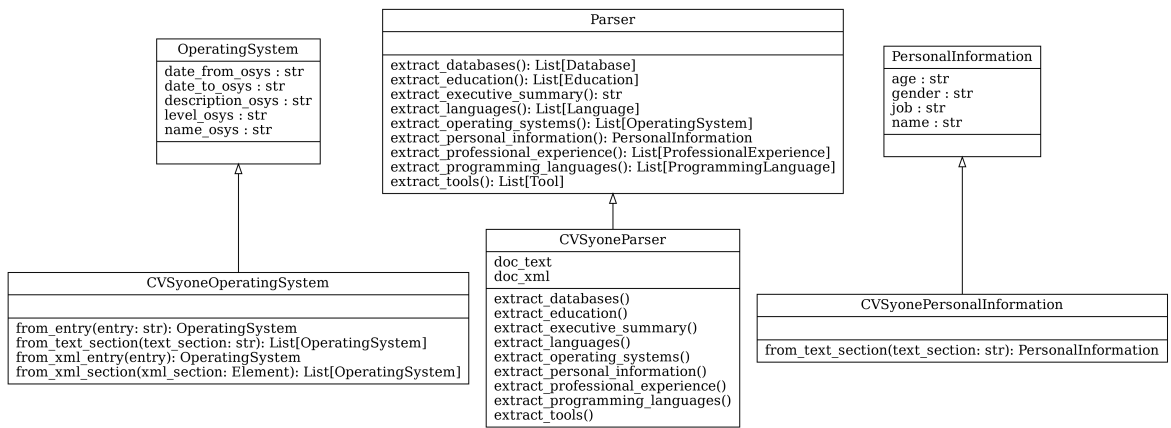


Figure 85: Knowledge extractor engine’s class diagram (part 2)

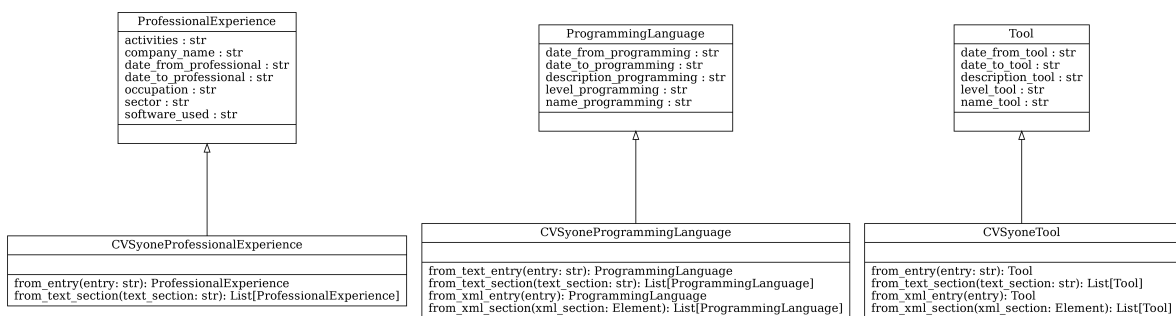


Figure 86: Knowledge extractor engine’s class diagram (part 3)

4.3 DOCUMENT BUILDER

Finally, the DocumentBuilder class was created to produce the structured document (in this case, a JSON document) corresponding to each CV. This is done by traversing the internal document representation and merging all sections of the CV together.

The structured document corresponding to the sample CV shown in Appendix B, which is returned by the document builder engine, can be seen in Appendix D.

The class diagram of this engine is shown in Figure 87.

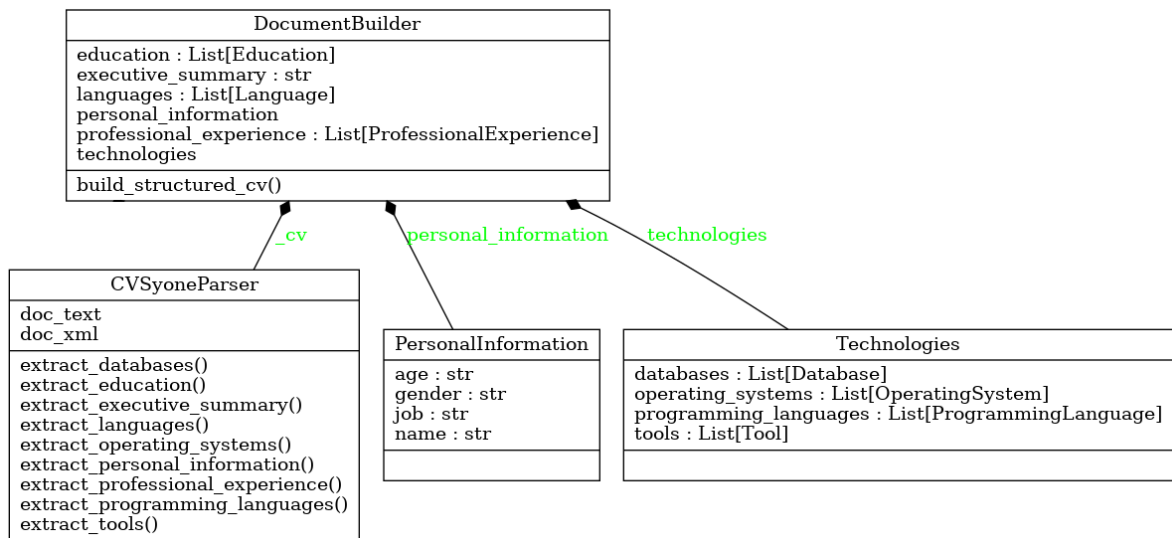


Figure 87: Document builder engine’s class diagram

4.4 SUMMARY

In this fourth chapter, the engines involved in processing CVs mentioned in the previous chapter (text extractor, XML extractor, knowledge extractor and document builder) were explained along with the obstacles that arose with their development. As such, it is possible to proceed to the description and discussion of the algorithms designed and implemented to achieve the first approach also presented in Chapter 3.



---

## SIMILARITY ALGORITHM (APPROACH A)

---

In this chapter of the dissertation, the engine that compares the CV and the job offer in the first approach presented in Chapter 3 is explained, starting by the elucidation of the general coding approach and following with the presentation of the techniques attempted and the respective results which, in turn, are also discussed.

### 5.1 IMPLEMENTATION AND RESULTS

Let us start by showing the application of USE with cosine similarity to unprocessed documents. Please, bear in mind that, for the other techniques that were applied, the methodology changes very little.

First of all, all of the libraries, modules and functions that will be needed are imported.

```
1 import pandas as pd
import tensorflow_hub as hub
3 from sklearn.metrics.pairwise import cosine_similarity
from utils import get_cvs, get_jobs, highlight_max
5 from syextractor.extractors.text_extractor import TextExtractor
```

Listing 5.1: Import libraries, modules and functions

Then, the Universal Sentence Encoder's TF Hub module is loaded.

```
module_url = "https://tfhub.dev/google/universal-sentence-encoder/4"
2 model = hub.load(module_url)
```

Listing 5.2: Load the USE's TF Hub module

After that, the previously defined model is applied to each job offer's text in order to obtain their respective word embeddings. Then, the same process is repeated for each CV. Next, the cosine similarity is applied to those word embeddings to calculate the affinity

between them and the results are saved into a dataframe. Finally, the best CV for each job offer is highlighted.

```

1 row_values = {}
  data = []
3
4 for job in get_jobs():
5     job_vec = model([TextExtractor("../data/job_offers/"+job).get_content()])
6     for cv in get_cvs():
7         cv_vec = model([TextExtractor("../data/cvs/"+cv).get_content()])
8         row_values.update({cv: cosine_similarity(job_vec, cv_vec)[0][0]})
9     data.append(row_values.copy())
11
12 df = pd.DataFrame(data, index = get_jobs())
13
14 df.style.highlight_max(color = 'lightgreen', axis = 1)

```

Listing 5.3: General code approach to apply the algorithm and the similarity measure (in this case, USE with cosine similarity)

Having the general approach when it comes to code, it is now possible to apply the various algorithms explored in Section 2.3 to the existing sample data and evaluate the respective results, so that the one that gives the most accurate outcome can be chosen. It should be noted that the sample to be taken into account contains twenty job offers and one hundred and sixteen CVs, which means that, of the algorithms explored, it was not possible to apply Doc2Vec, since this algorithm needs to be trained and there is a limited amount of data. Also note that the candidates presented throughout this document have been anonymized, in order to ensure their privacy.

First of all, Jaccard Similarity, TF-IDF, BERT and USE were applied to the whole sample without any preprocessing, in order to find the most adequate CV for each job offer. Note that, TF-IDF was applied using the scikit-learn library (*skl*). Besides that, take into account that TF-IDF, BERT and USE are vectorization algorithms, therefore, after the vectorization of the CV and the job offer in question, it was necessary to apply a similarity measure to compute the affinity between the vectors (in this case, cosine similarity, which was also applied using the scikit-learn library). Lastly, bear in mind that there already are BERT models tuned to be used for sentence/text embedding generation, so, initially, the bert-base-nli-mean-tokens model (*ber*) from SentenceTransformers library (*Sen*) was applied. Like with BERT, there are also pre-trained Universal Sentence Encoder models available, so TF2.0 Saved Model (v4) (*Ten, a*) from TensorFlow Hub (*Ten, b*) was employed.

The CVs chosen for each job offer based on the application of these algorithms can be found in Table 2. Analyzing the results of this table, it appears that, for the same job offer, the same CV was rarely chosen by the different algorithms.

Job offer	Jaccard Similarity	TF-IDF	BERT (bert-base-nli-mean-tokens)	USE
Backend Developer GO	Sharon Rodgers	Daniel Jacobson	David Johnson	Jennifer Ortiz
Backend Developer	Sharon Rodgers	Brenda Lynch	Tyler Garza	Richard Brady
BE Developer	Amanda Johnson	Brenda Lynch	Katherine Norris	Troy Hobbs
Business Development Manager	Edward Walters	Sean Smith	Edward Walters	Bethany Guerrero
Frontend Developer (Angular)	Carolyn Hall	Daniel Wilcox	Sharon Rodgers	Jennifer Ortiz
Frontend Developer (Vue.js) (1)	Robert Harris	Brenda Lynch	Sharon Rodgers	Jennifer Ortiz
Frontend Developer (Vue.js) (2)	Carolyn Hall	Brenda Lynch	Sharon Rodgers	Jennifer Ortiz
Infrastructure Architect and Pre-Sales	Edward Walters	Sean Smith	Rachel Tucker	Donald Shaffer
IT Recruiter	William Thomas	Benjamin Patel	Benjamin Patel	Bethany Guerrero
Junior Developers (1)	Daniel Wilcox	David Johnson	Katherine Norris	Patrick Carter
ML/Data Engineer	Mary Roy	Thomas Dixon	Tyler Garza	Thomas Dixon
.NET Developer (1)	Robert Harris	Daniel Wilcox	Katherine Norris	Brandy Taylor
.NET Developer (2)	William Thomas	Daniel Wilcox	Vincent Carrillo	Ann Hicks
QA Engineer	Edward Walters	Katherine Bullock	Tyler Garza	Katherine Bullock
Scrum Master (1)	Edward Walters	Sean Smith	Edward Walters	Donald Shaffer
Scrum Master (2)	Edward Walters	Brenda Lynch	David Johnson	Edward Walters
Solution Designer - Hardware	Carrie Yodes	Sean Smith	Carrie Yodes	Bethany Guerrero
Junior Developers (2)	Daniel Wilcox	David Johnson	Katherine Norris	Patrick Carter
System Administrator	Daniel Wilcox	Sarah Nichols	Christopher Christian	Sarah Nichols
Tech Lead/Senior Frontend	Daniel Wilcox	Ashley Moss	Tyler Garza	Ashley Moss

Table 2: CVs chosen for each job offer based on the application of the algorithms

As the candidate chosen for each job offer varies greatly from algorithm to algorithm, it was tried to cross the 5 best CVs chosen by each algorithm for each job offer, in order to verify if there were candidates in common. The result of crossing the 5 best CVs chosen by the various algorithms for each job offer can be found in the Table 3, Table 4, Table 5, Table 6, Table 7, Table 8, Table 9, Table 10, Table 11, Table 12, Table 13, Table 14, Table 15, Table 16, Table 17, Table 18, Table 19, Table 20, Table 21 and Table 22. It can be seen that some algorithms do have some candidates in common for certain job offers. However, those intersections still vary a lot between each other.

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity			Sharon Rodgers, Carrie Yodes	
TF-IDF				Thomas Dixon
BERT				
USE				

Table 3: Intersection of the top 5 CVs chosen by the various algorithms for the Backend Developer GO job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity		Carrie Yodes	Sharon Rodgers, Carrie Yodes	
TF-IDF			Carrie Yodes	Brenda Lynch, Jennifer Ortiz
BERT				
USE				

Table 4: Intersection of the top 5 CVs chosen by the various algorithms for the Backend Developer job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity			Robert Harris	
TF-IDF				Thomas Dixon, Troy Hobbs, Brenda Lynch, Jennifer Ortiz
BERT				
USE				

Table 5: Intersection of the top 5 CVs chosen by the various algorithms for the BE Developer job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity			Edward Walters	
TF-IDF			Wendy Brown	Bethany Guerrero
BERT				
USE				

Table 6: Intersection of the top 5 CVs chosen by the various algorithms for the Business Development Manager job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity			Sharon Rodgers	
TF-IDF			Ashley Moss	Troy Hobbs, Brenda Lynch
BERT				
USE				

Table 7: Intersection of the top 5 CVs chosen by the various algorithms for the Frontend Developer (Angular) job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity		Carrie Yodes	Carrie Yodes	Carrie Yodes
TF-IDF			Carrie Yodes	Ashley Moss, Brenda Lynch, Carrie Yodes
BERT				Carrie Yodes
USE				

Table 8: Intersection of the top 5 CVs chosen by the various algorithms for the Frontend Developer (Vue.js) (1) job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity			Sharon Rodgers	
TF-IDF			Ashley Moss	Brenda Lynch
BERT				
USE				

Table 9: Intersection of the top 5 CVs chosen by the various algorithms for the Frontend Developer (Vue.js) (2) job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity		Carrie Yodes	Edward Walters, Carrie Yodes	Carrie Yodes
TF-IDF			Carrie Yodes	Carrie Yodes
BERT				Carrie Yodes
USE				

Table 10: Intersection of the top 5 CVs chosen by the various algorithms for the Infrastructure Architect and Pre-Sales job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity		Benjamin Patel	Brandy Taylor, Benjamin Patel	Brandy Taylor, Benjamin Patel
TF-IDF			Benjamin Patel	Benjamin Patel
BERT				Brandy Taylor, Benjamin Patel
USE				

Table 11: Intersection of the top 5 CVs chosen by the various algorithms for the IT Recruiter job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity		Daniel Wilcox	Sharon Rodgers	
TF-IDF				
BERT				
USE				

Table 12: Intersection of the top 5 CVs chosen by the various algorithms for the Junior Developers (1) job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity			Vanessa Ferguson	
TF-IDF				Thomas Dixon, Patrick Carter, Christopher Jimenez
BERT				
USE				

Table 13: Intersection of the top 5 CVs chosen by the various algorithms for the ML/Data Engineer job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity			Sharon Rodgers, Robert Harris	
TF-IDF				Virginia Berry
BERT				
USE				

Table 14: Intersection of the top 5 CVs chosen by the various algorithms for the .NET Developer (1) job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity				
TF-IDF				Troy Nguyen
BERT				
USE				

Table 15: Intersection of the top 5 CVs chosen by the various algorithms for the .NET Developer (2) job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity		Katherine Bullock		Katherine Bullock
TF-IDF				Katherine Bullock, Nicholas Perkins, Sean Smith, Brenda Lynch
BERT				
USE				

Table 16: Intersection of the top 5 CVs chosen by the various algorithms for the QA Engineer job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity		Sara Larson, Sean Smith	Edward Walters	Jeffrey Edwards, Sean Smith
TF-IDF				Sean Smith
BERT				
USE				

Table 17: Intersection of the top 5 CVs chosen by the various algorithms for the Scrum Master (1) job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity		Edward Walters, Stephen Campbell	Edward Walters	Edward Walters
TF-IDF			Edward Walters	Edward Walters, Sean Smith
BERT				Edward Walters
USE				

Table 18: Intersection of the top 5 CVs chosen by the various algorithms for the Scrum Master (2) job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity		Wendy Brown, Carrie Yodes	Wendy Brown, Carrie Yodes	
TF-IDF			Wendy Brown, Carrie Yodes	
BERT				
USE				

Table 19: Intersection of the top 5 CVs chosen by the various algorithms for the Solution Designer - Hardware job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity		Daniel Wilcox	Sharon Rodgers	
TF-IDF				
BERT				
USE				

Table 20: Intersection of the top 5 CVs chosen by the various algorithms for the Junior Developers (2) job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity				
TF-IDF			Sharon Rodgers	Sarah Nichols
BERT				
USE				

Table 21: Intersection of the top 5 CVs chosen by the various algorithms for the System Administrator job offer

Algorithm	Jaccard Similarity	TF-IDF	BERT	USE
Jaccard Similarity		Daniel Wilcox	Sharon Rodgers	
TF-IDF			Ashley Moss	Ashley Moss, Jennifer Ortiz, Carolyn Hall
BERT				Ashley Moss
USE				

Table 22: Intersection of the top 5 CVs chosen by the various algorithms for the Tech Lead/Senior Frontend job offer



However, it was found that the bert-base-nli-mean-tokens model is obsolete and, therefore, produces low quality sentence embeddings. Thus, it was necessary to apply another model. For that, the all-mpnet-base-v2 (all, a) model (also from SentenceTransformers library) was chosen, since it has the highest average performance according to (Pre).

The CVs chosen for each job offer based on the application of this model can be found in Table 23. Analyzing the results of this table, it can be seen that, for each job offer, each CV chosen is different when compared to the previous model, except for the Scrum Master (1) job offer.

Job offer	BERT (all-mpnet-base-v2)
Backend Developer GO	Andrew Long
Backend Developer	Tiffany Orozco
BE Developer	Sharon Rodgers
Business Development Manager	Sean Smith
Frontend Developer (Angular)	Tiffany Orozco
Frontend Developer (Vue.js) (1)	Tiffany Orozco
Frontend Developer (Vue.js) (2)	Tiffany Orozco
Infrastructure Architect and Pre-Sales	John Richard
IT Recruiter	Roger Barrett
Junior Developers (1)	Tiffany Orozco
ML/Data Engineer	Thomas Dixon
.NET Developer (1)	Troy Nguyen
.NET Developer (2)	Ann Hicks
QA Engineer	Katherine Bullock
Scrum Master (1)	Edward Walters
Scrum Master (2)	Edward Walters
Solution Designer - Hardware	Hannah Santos
Junior Developers (2)	Tiffany Orozco
System Administrator	Cynthia Williams
Tech Lead/Senior Frontend	Tiffany Orozco

Table 23: CVs chosen for each job offer based on the application of the all-mpnet-base-v2 model

Despite being the one with the best average performance, the all-mpnet-base-v2 model is also one of the slowest, as expected. Thus, the all-MiniLM-L6-v2 (all, b) model (also from SentenceTransformers library) was applied as well, since it is one of the models with the best average performance/speed ratio (also according to (Pre)).

The CVs chosen for each job offer based on the application of this model can be found in Table 24. Analyzing the results of this table, it can be seen that, for each job offer, each CV chosen is different when compared to the previous two models, except for the Frontend Developer (Angular) job offer which has the same candidate chosen for this and for the all-mpnet-base-v2 model.

<b>Job offer</b>	<b>BERT (all-MiniLM-L6-v2)</b>
Backend Developer GO	Tyler Garza
Backend Developer	Ashley Moss
BE Developer	Jennifer Ortiz
Business Development Manager	Melissa Thompson
Frontend Developer (Angular)	Tiffany Orozco
Frontend Developer (Vue.js) (1)	Ashley Moss
Frontend Developer (Vue.js) (2)	Ashley Moss
Infrastructure Architect and Pre-Sales	Carolyn Hall
IT Recruiter	Katherine Turner
Junior Developers (1)	Garret Anderson
ML/Data Engineer	Jennifer Ortiz
.NET Developer (1)	Katherine Turner
.NET Developer (2)	Katherine Turner
QA Engineer	Jennifer Ortiz
Scrum Master (1)	Jennifer Ortiz
Scrum Master (2)	Jessica Thompson
Solution Designer - Hardware	Terry Medina
Junior Developers (2)	Garret Anderson
System Administrator	Roger Barrett
Tech Lead/Senior Frontend	Jennifer Ortiz

Table 24: CVs chosen for each job offer based on the application of the all-MiniLM-L6-v2 model

In addition to the experiments performed with cosine similarity, the remaining similarity measures mentioned in Section 2.4 were also applied, in order to verify if there was any change in the results.

Therefore, first, the USE algorithm with Euclidean distance was applied and the respective results can be seen in Table 25. From its analysis, it is observable that there are no changes when compared to the table with the results obtained from the application of the same algorithm with cosine similarity.

<b>Job offer</b>	<b>USE</b>
Backend Developer GO	Jennifer Ortiz
Backend Developer	Richard Brady
BE Developer	Troy Hobbs
Business Development Manager	Bethany Guerrero
Frontend Developer (Angular)	Jennifer Ortiz
Frontend Developer (Vue.js) (1)	Jennifer Ortiz
Frontend Developer (Vue.js) (2)	Jennifer Ortiz
Infrastructure Architect and Pre-Sales	Donald Shaffer
IT Recruiter	Bethany Guerrero
Junior Developers (1)	Patrick Carter
ML/Data Engineer	Thomas Dixon
.NET Developer (1)	Brandy Taylor
.NET Developer (2)	Ann Hicks
QA Engineer	Katherine Bullock
Scrum Master (1)	Donald Shaffer
Scrum Master (2)	Edward Walters
Solution Designer - Hardware	Bethany Guerrero
Junior Developers (2)	Patrick Carter
System Administrator	Sarah Nichols
Tech Lead/Senior Frontend	Ashley Moss

Table 25: CVs chosen for each job offer based on the application of USE with Euclidean distance

Then, the same algorithm was applied with Manhattan distance and the corresponding results can be seen in Table 26. Although there are many results in common when compared with the application of this algorithm with cosine similarity and Euclidean distance, it appears that the tables are no longer identical.

Job offer	USE
Backend Developer GO	Jennifer Ortiz
Backend Developer	Brenda Lynch
BE Developer	Troy Hobbs
Business Development Manager	Bethany Guerrero
Frontend Developer (Angular)	Jennifer Ortiz
Frontend Developer (Vue.js) (1)	Jennifer Ortiz
Frontend Developer (Vue.js) (2)	Jennifer Ortiz
Infrastructure Architect and Pre-Sales	Donald Shaffer
IT Recruiter	Bethany Guerrero
Junior Developers (1)	Virginia Berry
ML/Data Engineer	Thomas Dixon
.NET Developer (1)	Ann Hicks
.NET Developer (2)	Ann Hicks
QA Engineer	Katherine Bullock
Scrum Master (1)	Donald Shaffer
Scrum Master (2)	Edward Walters
Solution Designer - Hardware	Donald Shaffer
Junior Developers (2)	Virginia Berry
System Administrator	Sarah Nichols
Tech Lead/Senior Frontend	Jennifer Ortiz

Table 26: CVs chosen for each job offer based on the application of USE with Manhattan distance

Finally, this algorithm was applied with Chebyshev distance and the respective results can be seen in Table 27. It is clear that this table is the one that most differs from the others, with the exception of a few cases.

Job offer	USE
Backend Developer GO	Sean Smith
Backend Developer	Jeffrey Edwards
BE Developer	Patrick Carter
Business Development Manager	Donald Shaffer
Frontend Developer (Angular)	Richard Brady
Frontend Developer (Vue.js) (1)	Richard Brady
Frontend Developer (Vue.js) (2)	Richard Brady
Infrastructure Architect and Pre-Sales	Donald Shaffer
IT Recruiter	Patrick Carter
Junior Developers (1)	Patrick Carter
ML/Data Engineer	Thomas Dixon
.NET Developer (1)	Patrick Carter
.NET Developer (2)	Michael Clark
QA Engineer	Richard Brady
Scrum Master (1)	Sean Smith
Scrum Master (2)	Donald Shaffer
Solution Designer - Hardware	Jennifer Hicks
Junior Developers (2)	Patrick Carter
System Administrator	Jessica Thompson
Tech Lead/Senior Frontend	Richard Brady

Table 27: CVs chosen for each job offer based on the application of USE with Chebyshev distance

After all these experiments with the entire sample of CVs and job offers without any preprocessing, it was decided to do some tests with each of the sections of the CVs and the job offers in their entirety, in order to verify if there were any changes in the results. For this, the main reasoning was to apply the Jaccard Similarity to each section of the CV and to the entire job offer and, then, calculate the average of these results which, in turn, brings the similarity between the CV and the job offer in question. Then, these calculations were applied to the first job offer and to each CV and it was found that the results are different (with this approach the chosen candidate is Steven Moreno and with the previous approach the chosen candidate is Sharon Rodgers, as seen in Table 2).

In addition to the previous experience, another approach where the documents were preprocessed was also tried, in order to verify if there was any change in the results. The preprocessing consisted essentially of six steps (which were applied using the re module and the spaCy library):

- Normalization of line endings and extra white space removal;
- Tokenization;

- Stop words removal (which consists in removing the words that occur commonly across all the documents in the corpus, such as articles and pronouns which, in turn, have no significance in some of the NLP tasks (Hardeniya et al., 2016));
- Lemmatization;
- Lower casing;
- Punctuation removal.

The CVs chosen for each job offer based on the application of TF-IDF with cosine similarity to the preprocessed sample data can be found in Table 28. Comparing these results with the results of Table 2, it is possible to verify that there are only four matches between job offers and CVs in common.

Job offer	TF-IDF
Backend Developer GO	Eileen Smith
Backend Developer	Jennifer Ortiz
BE Developer	Brenda Lynch
Business Development Manager	Donald Shaffer
Frontend Developer (Angular)	William Thomas
Frontend Developer (Vue.js) (1)	Ashley Moss
Frontend Developer (Vue.js) (2)	William Thomas
Infrastructure Architect and Pre-Sales	Sara Larson
IT Recruiter	Wendy Coleman
Junior Developers (1)	Brenda Lynch
ML/Data Engineer	Thomas Dixon
.NET Developer (1)	Troy Nguyen
.NET Developer (2)	Troy Nguyen
QA Engineer	Katherine Bullock
Scrum Master (1)	Edward Walters
Scrum Master (2)	Edward Walters
Solution Designer - Hardware	Melissa Thompson
Junior Developers (2)	Brenda Lynch
System Administrator	Sharon Rodgers
Tech Lead/Senior Frontend	Ashley Moss

Table 28: CVs chosen for each job offer based on the application of TF-IDF to the preprocessed sample

The CVs chosen for each job offer based on the application of USE with cosine similarity to the preprocessed sample data can be found in Table 29. This table shows that the results of applying this technique are completely different from the results presented in Table 2.

Job offer	USE
Backend Developer GO	Garret Anderson
Backend Developer	Luis Baxter
BE Developer	William Thomas
Business Development Manager	Edward Walters
Frontend Developer (Angular)	Carolyn Hall
Frontend Developer (Vue.js) (1)	Carolyn Hall
Frontend Developer (Vue.js) (2)	Carolyn Hall
Infrastructure Architect and Pre-Sales	Wendy Coleman
IT Recruiter	Brandy Taylor
Junior Developers (1)	William Thomas
ML/Data Engineer	Timothy Lee
.NET Developer (1)	Robert Harris
.NET Developer (2)	Terry Medina
QA Engineer	Pamela Gutierrez
Scrum Master (1)	Ronald Cooper
Scrum Master (2)	Sean Smith
Solution Designer - Hardware	Benjamin Patel
Junior Developers (2)	William Thomas
System Administrator	Cynthia Williams
Tech Lead/Senior Frontend	William Thomas

Table 29: CVs chosen for each job offer based on the application of USE to the preprocessed sample

Note that this approach was applied only to TF-IDF and USE, as these algorithms are the ones that choose the most candidates in common, as seen above.

At this point, it is possible to conclude that it is not feasible to choose a particular technique among those presented, given the discrepancy observed in the results and the lack of plausibility in them (in fact, most candidates chosen for the different job offers according to the different techniques made no sense).

## 5.2 SUMMARY

In this chapter, the general coding approach to apply the similarity algorithms explored in Section 2.3 (with the exception of Doc2Vec) and the similarity measures explored in Section 2.4, in order to achieve the first approach described in Chapter 3 was explained and the results of their application were presented and discussed. At the end of this study, it was concluded that it was not possible to choose any of the techniques attempted, mainly because the results were not reasonable. That being said, let us move on to the description and discussion of the algorithm designed and implemented to achieve the second approach presented in Chapter 3 as well.

---

## SIMILARITY ALGORITHM (APPROACH B)

---

In this chapter, the engine that compares the CV and the job offer in the second approach presented in Chapter 3 is explained and the results of its application are discussed.

### 6.1 IMPLEMENTATION AND RESULTS

In this approach, and similarly to the CVs, it was also necessary to extract a structured document from each job offer. However, unlike the CVs, the existing job offers do not follow a pattern and, for that reason, it was not possible to follow the approach described in Section 4.1, Section 4.2 and Section 4.3. Instead, it was necessary to create a form so that the HR team could fill in the necessary information regarding the job offer and, from there, the respective structured document could be generated. For instance, suppose the HR team filled out the form based on the job offer from Appendix A. In this case, the corresponding structured document would be the JSON document from Appendix C.

Having the form, it was necessary to create a logic to calculate the percentage of similarity between a job offer and a CV. This logic involved calculating the score obtained by the candidate in each section (in this case, Professional Experience, Education, Languages and Technologies) and also calculating the maximum score that the candidate could obtain in those same sections. After that, the first score is divided by the latter, in order to give the percentage obtained by the candidate in each section. Then, each value is multiplied by the respective weight assigned to the section in question (since a section can have a different importance according to the job offer) and, finally, the sum of each product results in such percentage of similarity between a job offer and a CV.

That said, it is important to mention that the reasoning for calculating the score obtained by the candidate in each section is common to all sections, since it is always subdivided into two other calculations: calculation of the score obtained by each professional experience/education/language/technology found and calculation of the score obtained by each professional experience/education/language/technology not found.



Thus, and starting with the Professional Experience section, it is worth recalling that, although a job offer may ask for experience in a specific role (such as Machine Learning Engineer), other roles may be related and, therefore, also valuable (such as Data Engineer, Data Scientist, *etc.*). As a consequence, it was found useful to have a document for each related role, that is, for the Machine Learning Engineer role there will be a document with those keywords, plus Data Engineer, Data Scientist, *etc.*, for the IT Recruiter role there will be a document with those keywords, plus Head of Human Resources, *etc.* and so on. Furthermore, it was concluded that the score based on the number of years a candidate has held a role should increase towards a limit, not only because the difference between 1 and 5 years should be much more significant than the difference between 10 and 15 years, for example, since there is a point where there is not much more to learn, but also because that limit is necessary to find the maximum score that the candidate can obtain in this section, since there is not a maximum number of years of experience a person can have. The function in question and the respective graph can be seen below. Notice how the curve grows exponentially until approximately  $x = 10$  and, from there, it becomes much less steep, getting closer and closer to the value 25.

$$f(x) = 25 \left( 1 - \frac{1}{x+1} \right) \quad (9)$$

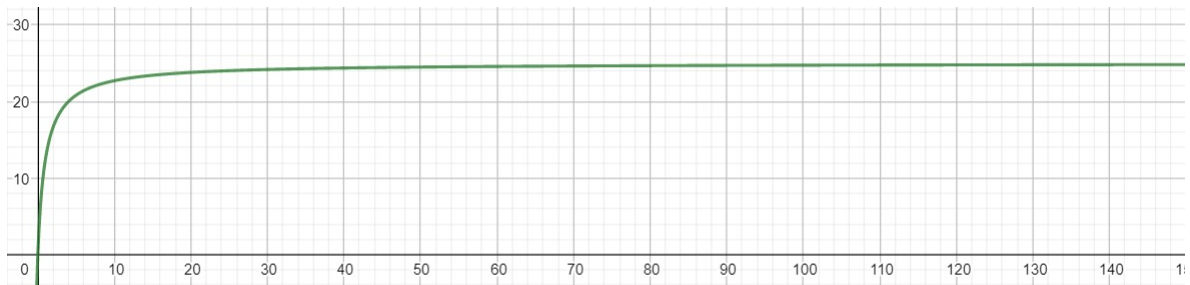


Figure 88: Graph of math function that returns score based on number of years

That said, regarding the calculation of the score obtained for each professional experience found, for each Professional Experience entry of the job offer, the respective document related to the role in question is opened and if any of the keywords existing in that document is found in any Professional Experience entry of the CV, one of the following applies:

- If the duration field exists in the job offer and the duration field in the CV (which is calculated using the start date and the end date with the help of the dateparser library and the dateutil module (`dat`, `c`)) is equal to or greater than the required duration, 1 is added to the value resulting from subtracting the duration in the job offer to the duration in the CV and  $f(x)$  is applied to this result, so that the candidate is valued;

- If the duration field exists in the job offer and the duration field in the CV is less than the required duration, the symmetric of  $f(x)$  is applied to the absolute value resulting from subtracting the duration in the job offer to the duration in the CV, so that the candidate is devalued;
- If the duration field does not exist in the job offer,  $f(x)$  is applied to the duration in the CV, so that the candidate is valued (since, in this case, all experience must be valued).

On the other hand, with regard to the calculation of the score obtained for each professional experience not found, a methodology similar to that explained above is applied to verify if any role of any Professional Experience entry of the job offer (or similar) is not found in any Professional Experience entry of the CV and, if so, one of the following applies:

- If the duration field exists in the job offer, the symmetric of  $f(x)$  is applied to the value in that field, since, in this calculation, the candidate always needs to be devalued;
- If the duration field does not exist in the job offer, the candidate receives -10 points (because, as mentioned before, they have to be devalued).

Finally, when it comes to calculating the maximum score that the candidate could obtain in this section, since the calculation of the score obtained for each professional experience found always resorts to  $f(x)$ , that maximum score equals to 25 points (limit of the function) for each Professional Experience entry of the job offer.

Continuing with the Education section, it is worth recalling that, like with the Professional Experience section, although a job offer may ask for a certain education level in a specific field, other fields may be related and, therefore, also valuable. For that reason, the methodology involving documents with related fields was also used in this section. Furthermore, please note that a candidate may have more than one level of education for the same required field, *i.e.*, imagine that the job offer asks for a Bachelor Degree in Computer Science and the candidate not only satisfies this requirement, but also has a Master Degree in Computer Science. In this case, only the latter should be considered, as it not only meets the requirement, but also exceeds it (note that a Master Degree is a higher education level than a Bachelor Degree). Lastly, keep in mind that for each education level a numerical value was mapped, in order to be able to relativize the different levels and make calculations between them. Note that each numerical value represents the typical number of years the respective education level generally lasts plus the numerical value of the previous education level (Bachelor Degree was mapped to 3, Post-Graduation was mapped to 4 and so on).

That said, with regard to the calculation of the score obtained for each education found, for each Education entry of the job offer, a methodology similar to that explained for the Professional Experience section is applied to verify if any field of any Education entry of the job offer (or similar) is found in any Education entry of the CV and, if so, the maximum

level of education in that field is extracted from the CV. In this situation, one of the following applies:

- If the maximum education level in the CV is equal to or greater than the education level in the job offer and the candidate has finished the degree, 1 is added to the value resulting from subtracting the latter to the first, so that the candidate is valued;
- If the maximum education level in the CV is less than the education level in the job offer and the candidate has finished the degree, the latter is subtracted to the first, in order to devalue the candidate;
- If the maximum education level in the CV is equal to or greater than the education level in the job offer and the candidate has not finished the degree, the latter is subtracted to the first, so that the candidate is valued (but not as much as in the first case);
- If the maximum education level in the CV is less than the education level in the job offer and the candidate has not finished the degree, 1 is subtracted to the value resulting from subtracting the latter to the first, so that the candidate is devalued (even more than in the second case);

On the contrary, when it comes to the calculation of the score obtained for each education not found, for each Education entry of the job offer, a methodology similar to that explained for the Professional Experience section is applied to verify if any field of any Education entry of the job offer (or similar) is not found in any Education entry of the CV and, if so, the candidate receives the symmetric of the result of adding 1 to the education level requested in the job offer, in order to devalue the candidate (evidently, even more than a candidate who has a degree in the required field, but a lower education level).

At last, with regard to the calculation of the maximum score that the candidate could obtain in this section, similarly to the first case in the explanation of the calculation of the score obtained for each education found (which is the one that results in the highest score), for each Education entry of the job offer, 1 is added to the value resulting from subtracting the level of education requested to the maximum level of education (in this case, Doctorate Degree which, in turn, is equivalent to 10).

Proceeding with the Languages section, take into account that, in this case, there are only five possible language levels, so, for the same reasons pointed out for the Education section, a numerical value was also mapped to each of those levels (Basic was mapped to 1, Average was mapped to 2, Good was mapped to 3, Very Good was mapped to 4 and Native was mapped to 5). In addition, all CVs have a proficiency level regarding reading, writing and speaking, so the average of these three values was used as the language level of the CV, in order to be able to compare with the language level of the job offer.

That said, with regard to the calculation of the score obtained for each language found, it is verified whether the language for each Language entry of the job offer is found in any Language entry of the CV and, if so, one of the following applies:

- If the language is required and the language level in the CV is equal to or greater than the language level in the job offer, 1 is added to the value resulting from subtracting the latter to the first, so that the candidate receives 1 point in case they have the same language level, 2 points in case they have one language level above, *etc.* and, therefore, is valued;
- If the language is required and the language level in the CV is less than the language level in the job offer, the latter is subtracted to the first, so that the candidate receives -1 point in case they have one language level below, -2 points in case they have two language levels below, *etc.* and, so, is devalued;
- If the language is not required, 0.5 is added to the result of dividing the result of subtracting the language level in the job offer to the language level in the CV by 10, so that the candidate receives 0.4 if they have one language level below, 0.5 if they have the same language level, 0.6 if they have one language level above, *etc.*, so that the candidate is never devalued, but does not obtain a greater appreciation than in the first case.

In contrast, with regard to the calculation of the score obtained for each language not found, it is verified whether the language for each Language entry of the job offer is not found in any Language entry of the CV and, if so and if that language is required, the candidate receives the symmetrical of the language level required by the job offer, as they should only be devalued for the required languages that they do not have.

Finally, when it comes to the calculation of the maximum score that the candidate could obtain in this section, for each Language entry in the job offer, there are two possible situations:

- In case that language is required, a similar approach to the one explained in the first case of the calculation of the score obtained for each language found (which, for required languages, is the one that results in the highest score) is applied, that is, 1 is added to the value resulting from subtracting the language level required by the job offer to the maximum language level (in this case, and as explained before, Native);
- In case that language is not required, a similar approach to the one explained in the third case of the calculation of the score obtained for each language found is applied, *i.e.*, 0.5 is added to the result of dividing the result of subtracting the language level required by the job offer to the maximum language level by 10.

Last but not least, let us take a look at the Technologies section. First of all, this section is subdivided into several subsections (Programming Languages, Databases, Operating Systems and Tools) which, in turn, and as for each section, may also have a different importance according to the job offer and, therefore, must have weights assigned to them as well. On top of that, it is important to mention that all of those subsections are very similar to each other, as they all have analogous fields. Thus, both the calculation of the score obtained for each technology (programming language, database, operating system or tool) found and the calculation of the score obtained for each technology not found were done in such a generic way that they can be used by all of those subsections. Taking these two considerations into account, it is easy to see that the calculation of the score obtained by the candidate in this section will involve the calculation of the score obtained by each technology found and also the calculation of the score obtained by each technology not found (as expected), but, now, for each Technologies subsection and, then, multiplying these results by the respective weights mentioned above. In addition, in this case, there are only five possible technology levels, so, once again, for the same reasons pointed out for the Education section, a numerical value was also mapped to each of those levels (Basic was mapped to 1, Average was mapped to 2, Good was mapped to 3, Very Good was mapped to 4 and Advanced was mapped to 5). Furthermore, unlike the previously mentioned sections, each subsection of this section has not just one, but two fields that play a major role when it comes to calculating scores. These fields are related to the technology level and duration and, therefore, in this case, the score calculations will always take these two parcels into account. Moreover,  $f(x)$  was also used in this section for reasons analogous to those explained for the Professional Experience section.

Having that in mind, as to the calculation of the score obtained for each technology found, it is verified whether each entry in the Technologies subsection in question of the job offer is found in any entry in the Technologies subsection in question of the CV and, if so, one of the following applies:

- If the technology is required:
  - Regarding the score related to the technology level, there are two possible cases:
    - \* If the technology level in the CV is equal to or greater than the technology level in the job offer, 1 is added to the value resulting from subtracting the latter to the first, in order to value the candidate (as explained for the Languages section);
    - \* If the technology level in the CV is less than the technology level in the job offer, the latter is subtracted to the first, in order to devalue the candidate (once again, as explained for the Languages section);

- Regarding the score related to the duration, there are three possible cases (similar to those explained for the Professional Experience section):
  - \* If the duration field exists in the job offer and the duration field in the CV (which, once more, is calculated using the start date and the end date with the help of the dateparser library and the dateutil module) is equal to or greater than the required duration, 1 is added to the value resulting from subtracting the duration in the job offer to the duration in the CV and  $f(x)$  is applied to this result, so that the candidate is valued;
  - \* If the duration field exists in the job offer and the duration field in the CV is less than the required duration, the symmetric of  $f(x)$  is applied to the absolute value resulting from subtracting the duration in the job offer to the duration in the CV, so that the candidate is devalued;
  - \* If the duration field does not exist in the job offer,  $f(x)$  is applied to the duration in the CV, so that the candidate is valued (since, in this case, all experience must be valued).
- If the technology is not required:
  - Regarding the score related to the technology level, an analogous calculation to that explained for not required languages is applied here, so that the candidate always receives an appreciation, but not as much as if it was a required technology;
  - Regarding the score related to the duration, there are three possible cases:
    - \* If the duration field exists in the job offer and the duration field in the CV is equal to or greater than the required duration, just 0.5 is added to the value resulting from subtracting the latter to the first and  $f(x)$  is applied to this result, in order to value the candidate, but, once more, not as much as if it was a required technology;
    - \* If the duration field exists in the job offer and the duration field in the CV is less than the required duration, the candidate receives 0 points, so that they are not valued (since they do not satisfy the requirements regarding this field) nor devalued (since the technology in question is not required);
    - \* If the duration field does not exist in the job offer, 0.5 is multiplied by the result of applying  $f(x)$  to the duration in the CV, so that the candidate is valued, but, anew, not as much as if it was a required technology;

Meanwhile, as to the calculation of the score obtained for each technology not found, it is verified whether each entry in the Technologies subsection in question of the job offer is not found in any entry in the Technologies subsection in question of the CV and, if so and if

that technology is required (because, once again, the candidate should only be devalued for the required technologies that they do not have), one of the following applies:

- If the duration field exists in the job offer, the candidate receives the symmetric of the result of adding the technology level required by the job offer to the result of applying  $f(x)$  to the duration required by the job offer;
- If the duration field does not exist in the job offer, the candidate receives just the symmetric of the technology level required by the job offer.

Ultimately, regarding the calculation of the maximum score that the candidate could obtain in this section, for each entry of each Technologies subsection of the job offer, there are two possible situations:

- In case that technology is required, 25 points, which is the limit of the function always used in the calculation of the score obtained for each technology found regarding the score related to the duration, are added to the result of applying a similar approach to the one explained in the first case of the calculation of the score obtained for each technology found regarding the score related to the technology level, that is, 1 is added to the value resulting from subtracting the technology level required by the job offer to the maximum technology level (in this case, and as explained before, Advanced);
- In case that technology is not required, 25 points, which come from a similar scenario to the one explained above, are added to the result of applying a similar approach to the one explained in the calculation of the score obtained for each technology found regarding the score related to the technology level, that is, 0.5 is added to the result of dividing the result of subtracting the technology level required by the job offer to the maximum technology level by 10.

After that, all the maximum scores computed for each Technologies subsection have to be multiplied by the respective weights mentioned above (and, of course, all the results are added together to, then, give the maximum score that the candidate could obtain in the Technologies section).

Having the calculation of the score obtained by the candidate in each section and the calculation of the maximum score that the candidate could obtain in those same sections, it is now possible to calculate the percentage of similarity between a job offer and a CV, as explained at the beginning of this chapter, paying special attention to the case in which the maximum score that the candidate can obtain is 0 (which results in a percentage of 0, as it is not possible to divide by this value) and to the case in which the final calculation results in a negative value (which also results in a percentage of 0, because there can be no negative percentages).

At this stage, it is important to rank the candidates according to the job offer they applied for from best to worst according to their percentage of similarity and display only the best  $x$ ,  $x$  being a configurable parameter that is stored in a dictionary of settings along with the weights mentioned throughout this chapter (which, of course, are also configurable).

That said, applying the developed algorithm to the same one hundred and sixteen CVs referred to in Chapter 5 according to the job offer presented in Appendix A with the configurations shown below, leads to the results presented in Table 30.

```

1 configuration = {
2     "limit": 5,
3     "weights": {
4         "sections": {
5             "professional_experience": {"weight": 0.25},
6             "education": {"weight": 0.25},
7             "languages": {"weight": 0.25},
8             "technologies": {
9                 "weight": 0.25,
10                "subsections": {
11                    "programming_languages": {"weight": 0.25},
12                    "databases": {"weight": 0.25},
13                    "operating_systems": {"weight": 0.25},
14                    "tools": {"weight": 0.25},
15                },
16            },
17        },
18    },
19 }

```

Listing 6.1: Dictionary of settings used as an example

Candidate	Percentage of Similarity (2 d.p.)
Thomas Dixon	71.71%
Patrick Carter	58.21%
Ashley Lawrence	35.37%
Christopher Jimenez	28.66%
Melvin Fields	20.62%

Table 30: Top 5 candidates for the job offer presented in Appendix A

These results were examined both in terms of calculations and in terms of order and, from there, it was concluded that they are plausible. It should be noted that most of these candidates have experience in the field of Machine Learning, have a degree in Computer Science or similar, demonstrate fluency in English and satisfy some of the necessary technologies, which makes them more suitable for the job offer in question.



## 6.2 SUMMARY

In this sixth chapter, the similarity algorithm engine of the second approach presented in Chapter 3 was explained in detail. After testing the algorithm, it was concluded that the results were plausible and, therefore, it was feasible to solve the initial problem. At this point, it is finally possible to summarize all the work and present the conclusions achieved throughout this Master's Project.

---

## CONCLUSION

---

This Master's Project is entitled "Automation of companies' recruitment process: development of an algorithm capable of ranking CVs according to job offers" and was proposed by Syone SBS Software - Tecnologia e Serviços de Informática, S.A., since its Human Resources team (in particular, the recruitment crew) was facing an enormous work load when it came to screening CVs.

This Master's Project aims to prove the underlying research hypothesis "It is possible to develop an algorithm capable of automatically matching the most suitable applicants to a certain job offer". For that, several tasks were performed:

1. Analysis of different types of CVs (such as Europass, LinkedIn profile as a CV, Curriculum Lattes, CIÊNCIAVITAE and Syone's internal CV) and development of an ontology defining the information necessary to fully characterize a CV;
2. Study of the typical components (entities and relationships) and steps (sentence splitting, tokenization, POS tagging, lemmatization, NER and relationship extraction) involved in the Information Extraction process;
3. Exploration and comparison of the most popular similarity algorithms (Jaccard Similarity, TF-IDF, Doc2Vec, BERT and USE);
4. Research of some of the most popular similarity measures (cosine similarity, Euclidean distance, Manhattan distance and Chebyshev distance);
5. Description of the proposed architecture for the system;
6. Implementation of the first four engines in charge of processing the CVs and discussion of the problems that came along;
7. Application of the previously studied similarity algorithms (and similarity measures) as a first approach to calculate the affinity between a job offer and a CV;
8. Development of another approach to calculate this affinity.

In the following paragraphs, the aforementioned phases will be summarized and the challenges that have arisen and the conclusions drawn in each phase will be revisited. Finally, some suggestions that may be followed as future work will be presented.

Regarding the first phase, several CV formats in vogue were analyzed, in order to compare the information required by each one of them. It was found that, although there were certain fields and patterns in common (for instance, they all required some data concerning the work experience and education and, usually, the former came after the latter), some were better suited to a certain area of expertise and/or country than others. Bear in mind that, in addition to the CV formats mentioned, others were analyzed (such as the College Art Association's Visual Artist Format). However, it was found that, once again, these formats were very targeted to certain areas of expertise and these, in turn, were outside the Information Technology (IT) domain. Therefore, Syone was unlikely to receive such CVs and, so, they were not deeply explored or discussed. After that large research concerning CV sources and formats, an ontology based on Syone's internal CV was developed, in order to identify the information it was necessary to extract from each CV.

In terms of the second phase, it was necessary to explore the various components and stages of Information Extraction, as they could be necessary to extract information from CVs and job offers. First, the main type of structures extracted from an unstructured source (entities and relationships) were studied and it was concluded that entities are typically noun phrases, like proper names, organizations, *etc.* and that relationships are defined over two or more related entities, like "is employee of" between a person and an organization, *etc.*. Then, the typical workflow of IE was studied and it was found that it consists of sentence splitting (where the text is divided into sentences), tokenization (where each sentence is broken down according to a set of delimiters generating tokens), POS tagging (where a grammatical category is assigned to each token), lemmatization (where the various forms of a token are mapped to its root form), NER (where tokens are recognized as people, organizations, *etc.*) and relationship extraction (where relationships between entities are identified and collected). After that, it was found that NER approaches are classified into two main types, these being Knowledge Engineering (which uses domain knowledge of human expertise represented in a machine-understandable form) and automatic training (which uses ML algorithms), and that the first approach not only requires less effort to define patterns and develop rules when compared to the effort required to annotate a large amount of training data in the second approach, but also tends to produce a higher performance, since human expertise results in more accurate patterns. Finally, a Python library that provides the functionalities to perform all these steps called spaCy was also explored.

Moreover, when it comes to the third phase, the most popular similarity algorithms were explored and compared, in order to apply them to the CVs and job offers and, consequently, find the affinity between them. To summarize, it was found that Jaccard Similarity is defined

as the size of the intersection divided by the size of the union of the sample sets; TF-IDF measures how much a term is relevant to a document in a collection of documents; Doc2Vec is capable of transforming text documents into a vectorized form and is heavily based on Word2Vec; BERT is based on Transformers and generates contextualized embeddings from the input words and USE encodes text into high dimensional vectors. After comparing all these algorithms, it is clear that each of them has its pros and cons, but BERT stands out from the rest as it presents state-of-the-art results in a wide variety of NLP tasks and, therefore, has been widely used since its publication.

As to the last phase of the research part, the most popular similarity measures were investigated, in order to apply them to the vectorization algorithms mentioned before. To recap, the cosine similarity is measured by the cosine of the angle between two vectors; the Euclidean distance is calculated as the square root of the sum of the squared differences between two vectors; the Manhattan distance is computed as the sum of the absolute differences between two vectors and the Chebyshev distance is calculated as the greatest of the differences between two vectors along any coordinate dimension.

With respect to the fifth phase, an architecture for the system was proposed. In the proposed architecture, the system receives a CV which goes through a text extractor engine and/or a XML extractor engine to extract its text and/or XML, respectively. In parallel, it either receives a file corresponding to the job offer which, in this case, undergoes the same preprocessing or the user introduces the job offer's relevant information via an appropriate interface. Then, both the CV's information and the job offer's information go through a module that compares them and returns the respective affinity. To do so, two different approaches were tried, aiming at a clear understanding of the pros and cons of each one of them. In the first approach, regarding the job offer, the first path mentioned before is followed, which means that both the CV text and/or XML and the job offer text and/or XML go through an engine that runs a ML algorithm followed by a similarity measure, in order to compute the affinity between those two inputs. In the second approach, the CV text and/or XML passes through the knowledge extractor engine, so that it extracts its relevant information (Personal Information, Executive Summary, *etc.*) and, after that, the document builder engine merges all the extracted information to generate the corresponding structured document. In parallel, regarding the job offer, the second path mentioned before is followed, that is, the user fills in a form with the job offer's relevant information and the associated interface module returns the corresponding structured document. Lastly, the structured documents together with the rules that define the weights for each requirement are handed over to the similarity algorithm engine which, in turn, returns the affinity between the two documents.

Note that, from the sixth phase onwards, it was decided to deal only with Syone's CV format, but, in future iterations, other CV formats may be handled following a similar

approach. In this phase, the implementation of the text extractor, XML extractor, knowledge extractor and document builder engines (*i.e.*, CVs processing phase) was explained and the main challenges that came along the process were pointed out. First of all, the text extractor engine was created to extract the content of documents in plain text format while the XML extractor engine was created to extract XML from documents, allowing a more structured analysis. Then, regarding the knowledge extractor engine, the main logic to collect the information from each section involved the following steps:

- For the Personal Information, Executive Summary, Professional Experience, Education and Languages sections:
  - Extract the text from each section. This involved, first, extracting the text from the document using the text extractor engine and, then, finding the word that indicates the beginning of the section and the word that indicates the end of it and, finally, extracting the text between them;
  - Extract the fields from each section (such as name, job description, *etc.*). For the Personal Information section, the logic was similar to the one explained above. However, for the Professional Experience and Education sections, it was a little bit different, as they have multiple entries. Thus, each entry was divided on the keyword that indicates the beginning of it (in this case, "Dates (from to)") and, then, each field was extracted following an approach similar to the one used for the Personal information. Finally, the Languages section also has several entries, but, since it is a table, some patterns were found and made the extraction somewhat easier. It was noticed that its rows were separated by two newline characters (so the table was split on this delimiter, in order to get the entries) and, within each row, each field was separated by a newline character (so these were extracted by splitting each entry on that delimiter).
- For the sections referring to the Technologies table (Programming Languages, Databases, Operating Systems and Tools), since this table does not follow any pattern (unlike the Languages table), it was necessary to use the XML extractor to extract the XML corresponding to the table and, consequently, have access to the table's components and be able to do a much more structured manipulation.

Lastly, the document builder engine was created to produce the structured document corresponding to each CV which was done by, essentially, merging all sections extracted before together.

Bear in mind that this was one of the most challenging phases of the project, as it was necessary to overcome several obstacles inherent to natural language. In fact, many candidates had missing or duplicate fields which, in itself, was a big problem. In addition, the fact that some candidates filled in the fields incorrectly (such as typing a newline

character in the middle of a field's text) made it very difficult to extract information from the Technologies table. Finally, the biggest obstacle came with the date fields, as many candidates did not follow the same date format, did not fill in valid dates, did not specify the start/end date, *etc.*.

In the penultimate phase, before presenting the results from the application of the similarity algorithms previously studied, the general coding approach to achieve those results was briefly explained. After explaining that, those similarity algorithms were applied, with the exception of the Doc2Vec algorithm, since this one needs to be trained and there was a limited amount of data (twenty job offers and one hundred and sixteen CVs), *i.e.*, only Jaccard Similarity, TF-IDF, BERT and USE were applied. Note that the last three are vectorization algorithms and, therefore, it was necessary to apply the similarity measures also previously studied (cosine similarity, Euclidean distance, Manhattan distance and Chebyshev distance) to the generated vectors, in order to compute the affinity between them. That said, first of all, so that the most adequate CV for each job offer was found, Jaccard Similarity, TF-IDF (with cosine similarity), BERT (bert-base-nli-mean-tokens from SentenceTransformers library with cosine similarity) and USE (TF2.0 Saved Model (v4) from TensorFlow Hub with cosine similarity) were applied to the whole sample and it was observed that the same CV was rarely chosen by the different algorithms. After that, the five best CVs returned by these same algorithms for each job offer were crossed, in order to verify if there were candidates in common and, although this was verified for some cases, the majority of the intersections varied greatly from each other. Then, it was found that the bert-base-nli-mean-tokens model was obsolete, so the all-mpnet-base-v2 model (also from SentenceTransformers library), which has the highest average performance, was applied instead and the results were totally different when compared to the previous model (except for the Scrum Master (1) job offer). Despite being the one with the best average performance, this algorithm is also one of the slowest, so the all-MiniLM-L6-v2 model (also from SentenceTransformers library) was applied as well, since it is one of the models with the best average performance/speed ratio and the results were also different when compared to the previous two models (except for the Frontend Developer (Angular) job offer which has the same candidate chosen for this and for the preceding model). Next, to see if there was any change in the results, the remaining similarity measures (Euclidean distance, Manhattan distance and Chebyshev distance) were applied with the USE algorithm and it was seen that there were no changes with the first measure, there were some changes with the second measure and there were a lot of changes with the third one. After all these experiments with the unprocessed documents, in order to see if there were any changes in the results, it was decided to apply Jaccard Similarity to each section of each CV and to the first job offer in its entirety and, then, calculate the average of these results to find the similarity between them and it was verified that the chosen candidate for this and for the previous approach (Jaccard Similarity

with the unprocessed documents) is different. Additionally, and also to see if there were any changes in the results, TF-IDF with cosine similarity and USE with cosine similarity (which were the algorithms that chose the most candidates in common, according to their intersections) were applied to the preprocessed documents (this preprocessing consisted of normalization of line endings and extra white space removal, tokenization, stop words removal, lemmatization, lower casing and punctuation removal) and, while for the first algorithm there were four matches comparing with the same approach with the unprocessed documents, for the second algorithm the results were completely different. After all this, it was concluded that it was not possible to choose one of the techniques previously applied, because the results were very discrepant and most of the candidates chosen for each job offer by each algorithm were not suited for them.

On the other hand, in the last phase, a different approach was tried. Here, the main logic to compute the affinity between a job offer and a CV involved the following steps:

- Calculating the score obtained by the candidate in each section (which, in turn, is subdivided into the calculation of the score obtained by each professional experience/education/language/technology found and the calculation of the score obtained by each professional experience/education/language/technology not found);
- Calculating the maximum score that the candidate could obtain in those same sections;
- Dividing the former by the latter, in order to get the percentage obtained by the candidate in each section;
- Multiplying each value by the respective weight assigned to the section in question (which is configurable);
- Summing each product to obtain such percentage of similarity between a job offer and a CV.

Then, according to this percentage of similarity, the candidates were ranked from best to worst in relation to the job offer they applied for and only the best  $x$  were displayed ( $x$  also being a configurable parameter). Note that this approach takes advantage of the structured documents corresponding to the CVs and the job offers.

That said, the algorithm was tested, in particular, with the job offer presented in Appendix A and the one hundred and sixteen CVs that were made available, with the same weight for each section and subsection (in this case, 0.25) and a limit equal to 5. This way, the algorithm returned the 5 most suitable CVs for the job offer in question and, after analyzing the results, it was concluded that these were plausible choices (notice how the candidate who is in first place, whose CV, in turn, is found in the Appendix B, satisfies most of the requirements of the job offer).

Also notice that these results, effectively, make sense, unlike the previous approach where the results not only varied from technique to technique, but also returned candidates who were not suitable for the job offers in question. In particular, for the job offer in Appendix A, were returned candidates who have a background in DevOps, for example, which has nothing to do with the Machine Learning expertise required. These unfortunate results might be explained by the fact that the similarity algorithms used in the first approach deal with the documents as a whole, which means that certain requirements demanded by the job offer for a specific field may be found, but in the wrong field, which conducts to misleading results.

So, it is fair to conclude that the Master's Work here reported proved that it is possible to develop an algorithm capable of automatically matching the most suitable applicants to a certain job offer.

However, as future work, there are some details that, in my opinion, could be added to make this algorithm more robust. In the first place, candidates who present more skills than those requested should be valued. Furthermore, it would be interesting to give a negligible value to candidates who, despite not meeting the requirements in terms of programming languages, have experience in similar programming languages (for example, if the job offer asks for Java, but, instead, the candidate has Python, which can also be used for object-oriented programming, they should be valued). Additionally, it may be helpful to provide details on the attributes of the analyzed CV that match the requirements of the job offer. Moreover, it would be useful to develop a Graphical User Interface (GUI), so that users (in this case, the HR team) could interact with the algorithm in a much more intuitive and pleasant way. Lastly, although this Master's Project has been more directed towards Syone's CV format, in future iterations, and as said before, it would be interesting to adapt this work to other CV formats, such as Europass CVs, LinkedIn profiles as CVs, *etc.* and, consequently, being able to handle more and more formats.



# A

---

## SAMPLE JOB OFFER

---

This appendix shows a sample job offer.

# ML/Data Engineer

## Job description

We're looking for a Data engineer to join a major client!

## Requirements

### What do I need to bring?

- Proven experience as MLE/DE
- BSc in Computer Science or a similar field, with a very solid Spark and Python programming experience; Scala is a plus - for creating data pipelines that will fuel machine learning models in an Azure Databricks based environment
- Proven experience with the engineering aspects of popular machine learning practices, libraries, and platforms - such as the ones that allow serving and productionizing machine learning models
- Acquaintance with CD4ML and MLOps, software design patterns and RESTful APIs
- Main tech stack: Databricks, Spark, Airflow, MLflow, MLeap, Kafka, Delta Lake and Azure ADLS (Parquet, Avro), Azure Devops.
- Proficient in code versioning tools: Github
- Strong English communicator;
- Comfortable with an Agile methodologies (Scrum, Kanban) using Jira Agile

### What can Syone offer me?

- Integration in an organization with profound and sustained growth and involvement in pioneering projects with innovative technological solutions;
- Strong IT training plans;
- Professional evolution with intervention in ambitious technological projects, both national and internationally.

# B

---

## SAMPLE CV

---

This appendix shows a sample CV.

## PERSONAL INFORMATION

**NAME:** Thomas Dixon

**JOB DESCRIPTION:** Machine Learning Engineer

**AGE:** 30

**GENDER:** Male

## EXECUTIVE SUMMARY

The consultant is taking a PhD in Computer Vision in University of Coimbra and hold a master's degree in computer science and Business Management from ISCTE and a bachelor's degree in Computer Science and Engineering from Instituto Polytechnic of Castelo Branco. The consultant started the career in 2014 at IT Sector (Millenium BCP project) as a Java developer. In 2016, changed to GS1, to work with Outsystems, as fullstack developer. Between 2017 and 2019, assumed functions as Data Scientist and Machine Learning Engineer. The consultant was responsible for developing a Document classification system, where was able to obtain 70% of accuracy on a multi-label classification problem using TF-IDF and SVM and worked on an object detection module for automated document parsing (PDFs) using state of the art Deep Learning techniques. At Celfocus as a Machine Learning Engineer, was responsible for building and developing multi label classification models, building machine learning algorithms, creation of pipeline and a classification model for anomaly detection in Set up Boxes, using Isolation Forest. Regularly works with Graphs to solve problems like costumer churn and client segmentation. Currently, works at Syone managing and leading a team of machine learning composed by 5 members, present the work done to stakeholders other than that he is working on a retail client where was responsible to maintain and create new machine learning pipelines. The consultant gives python training and architecture new proposals. The consultant is a team-player and teammate, a good communicator, that enjoys apply to projects that everyone is committed working towards the same goal.

---

**Proprietary & Confidential:** This document is part of Syone S.A. intellectual property. It cannot be copied or submitted to other entities without Syone S.A. written consent.

**Syone – Tecnologia e Serviços de Informática, S.A.**

Rua Alfredo da Silva, nº 8ª - Edif. Stern, Piso 3-D, 2610-016 Alfragide

**t:** +351 21 424 67 10 **f:** +351 21 424 67 19 **w:** <http://www.syone.com> **nif:** 504 729 624

## PROFESSIONAL EXPERIENCE

Dates (from to)	May 2021 - Present
Company Name	Syone
Type of business or sector	Consultancy
Occupation or position held	Lead Machine learning Engineer
Main activities and responsibilities	<p>During this experience, the consultant was responsible for the following activities:</p> <ul style="list-style-type: none"> <li>Responsible for leading a team with 5 members on 2 different projects: customer churn in telecommunications and building an NLP module that classify sentiment and extract Entities from phrases;</li> <li>Responsible for Cognitive Board where specific tasks are assigned to each member of the team</li> <li>Responsible for the architecture of the previously modules and designing a REST API to interact with the modules and integrate with other services;</li> <li>Responsible for mentoring two team members during an internship where was created a scrapping and classification algorithms. The purpose was to create an algorithm that gives the best cv match against job offers;</li> <li>Responsible to maintain and add multiple machine learning models in production (Databricks);</li> <li>Development of NLP model to predict whether a review need to answer or not;</li> </ul>
Software and Environments used	<ul style="list-style-type: none"> <li>Python, Databricks, MLFlow, Azure Service Studio, FastAPI Jupyter notebooks, Jira, Gitlab.</li> </ul>

Dates (from to)	October 2019 – May 2021
Company Name	Celfocus
Type of business or sector	Telecommunications
Occupation or position held	Machine learning Engineer
Main activities and responsibilities	<p>During this experience, the consultant was responsible for the following activities:</p> <ul style="list-style-type: none"> <li>Responsible for building and developing multi label classification models to classify user requests. with multi hierarchical levels, three levels. The first level had 17 categories, we achieve a F-score of 78% on testing set. On the second level we achieve a F-Score of 60%. We use, TF-IDF, SVM, Naive Bayes, Random Forest, and KNN to create our models;</li> <li>Responsible for building unsupervised machine learning algorithms (Standard Deviation from average, MAD, Z-score, Grubbs), and the creation of pipeline for anomaly detection;</li> <li>Responsible for mentoring one team member in a time series data science project where we tested unsupervised models MAD, ARIMA, Isolation Forest;</li> <li>Responsible for creating new business value proposals and presenting them to stakeholders;</li> <li>Development of classification model to detect anomalies in in Set up Boxes using Isolation Forest.</li> </ul>
Software and Environments used	<ul style="list-style-type: none"> <li>Python, Scala, Apache Spark, Jira, Pyspark, AWS (S3, EMR), GitLab, Zeppelin, Jupyter Notebooks, Python Bokeh.</li> </ul>

Dates (from to)	September 2019 – Feb 2020
Company Name	ISCTE-IUL
Type of business or sector	Teaching
Occupation or position held	Instructor

Main activities and responsibilities	<p>During this experience, the consultant was responsible for the following activities:</p> <ul style="list-style-type: none"> <li>Responsible for create lecture contents and teaching hands-on classes on Big Data Algorithms.</li> </ul>
Software and Environments used	<ul style="list-style-type: none"> <li>Apache Spark</li> </ul>

Dates (from to)	May 2017 – October 2019
Company Name	Novabase
Type of business or sector	Finance
Occupation or position held	Data Scientist and Machine Learning Engineer
Main activities and responsibilities	<p>During this experience, the consultant was responsible for the following activities:</p> <ul style="list-style-type: none"> <li>Responsible for proposal creation, status report, in a python module for categorize, project management documents using Heuristics (specific keywords) and machine learning techniques (Data Extraction, Data cleaning, TF-IDF, SVM, RF, DT);</li> <li>Mentoring in a 1-month internship in a data science project whereas the main purpose is evaluating if the house prices will be increase or decrease;</li> <li>Create a parser library to convert PDF, Image, HTML files into Data Objects, Using Heuristics (Regex, Keywords) and Deep Learning for object detection localization using Faster R-CNN, SSD, YOLO networks;</li> <li>Bug fixing and code refactor on a document platform, which classify documents using machine learning techniques NLP.</li> </ul>
Software and Environments used	<ul style="list-style-type: none"> <li>Java, Python, Deep Learning, Image Processing, Tensorflow.</li> </ul>

Dates (from to)	April 2016 – May 2017
Company Name	GS1 Portugal
Type of business or sector	Retail
Occupation or position held	Full stack developer
Main activities and responsibilities	<p>During this experience, the consultant was responsible for the following activities:</p> <ul style="list-style-type: none"> <li>Developed a tracking platform system for food products that come from different countries and are mixed. e.g.: sausage production.</li> </ul>
Software and Environments used	<ul style="list-style-type: none"> <li>Outsystems, .Net, JavaScript, CSS, HTML5.</li> </ul>

Dates (from to)	April 2015 – April 2016
Company Name	KCSIT – Timwe Group
Type of business or sector	Telecommunications
Occupation or position held	Backend developer
Main activities and responsibilities	<p>During this experience, the consultant was responsible for the following activities:</p> <ul style="list-style-type: none"> <li>Developing Billing Api's integrated with Mobile Operators in order to charge clients when they by some content on our platform.</li> </ul>
Software and Environments used	<ul style="list-style-type: none"> <li>Java, Shell Scripting, REST, SOAP, SVN, JUnit, Tomcat Junit</li> </ul>

Dates (from to)	Jul 2014 – April 2015
Company Name	IT Sector – Millenium BCP
Type of business or sector	Finance
Occupation or position held	Backend developer



Main activities and responsibilities	<p>During this experience, the consultant was responsible for the following activities:</p> <ul style="list-style-type: none"> <li>• Incident Analysis in Mobile and Web Channels of the bank;</li> <li>• Developing new features and web services for web and mobile consumption. Ongoing maintenance.</li> </ul>
Software and Environments used	<ul style="list-style-type: none"> <li>• Java, Shell Scripting, SOAP, JUnit, TFS.</li> </ul>

## EDUCATION

Dates (from to)	2021 - present
Name and type of organization providing education and training	University of Coimbra
Title of qualification awarded	PhD in Eletrotechnical Engineering – Specialization in Computer Vision

Dates (from to)	2017-2020
Name and type of organization providing education and training	ISCTE-IUL – University of Lisbon
Title of qualification awarded	Master’s Degree in Computer Science and Business management

Dates (from to)	2010 – 2014
Name and type of organization providing education and training	IPOB Institute Polytechnic of Castelo Branco
Title of qualification awarded	B.S in Computer Science and Engineering

## LANGUAGES

Other Languages	Reading Skills	Writing Skills	Verbal Skills
Portuguese	Native	Native	Native
English	Very Good	Very Good	Very Good
Spanish	Basic	Basic	Basic
German	Basic	Basic	...

Language: Basic | Average | Good | Very Good | Native

## TECHNOLOGIES

Technology	Level	Duration	Date	Detail Description
Programming Languages				
Java	Good	3Y	2014-2016	N/A
Python	Good	4Y	2017-2022	N/A
Scala	Average	8M	2021-2021	N/A
Javascript	Basic	1Y	2017-2017	N/A
.NET	Basic	1Y6M	2016-2017	N/A
Outsystems	Average	1Y2M	2016-2017	N/A
HTML	Average	1Y8M	2016-2017	N/A
CSS	Average	1Y8M	2016-2017	N/A
Databases				
PostgreSQL	Good	2Y	2018-2019	N/A
Oracle	Good	2Y	2014-2016	N/A

MSSQL	Average	6M	2021-2021	N/A
Big data databases	Average	3 years	2018-2022	N/A
Operating Systems				
Linux (Ubuntu,	Good	6Y	N/A	N/A
Windows	Very Good	6Y	N/A	N/A
Tools				
Intellij Idea	Good	3Y	N/A	N/A
VS code	Good	2Y	N/A	N/A

**Level:** Basic | Average | Good | Very Good | Advanced

**Duration:** X Months | X Years

**Date:** Year(s) you worked with the technology (e.g. 2015 – 2016; 2021)

---

## JSON FILE CORRESPONDING TO THE SAMPLE JOB OFFER

---

This appendix shows the JSON file corresponding to the sample job offer presented in Appendix A.

```
1 {
2   "professional_experience": [
3     {
4       "duration": null,
5       "occupation": "MLE/DE"
6     }
7   ],
8   "education": [
9     {
10      "is_finished": true,
11      "area": "Computer Science",
12      "level": "BSc"
13    }
14  ],
15  "languages": [
16    {
17      "language": "English",
18      "level": "Very Good",
19      "is_required": true
20    }
21  ],
22  "technologies": {
23    "programming_languages": [
24      {
25        "name_programming": "Python",
26        "level_programming": "Very Good",
27        "duration_programming": null,
```

```
28         "is_required": true
29     },
30     {
31         "name_programming": "Scala",
32         "level_programming": "Good",
33         "duration_programming": null,
34         "is_required": false
35     }
36 ],
37 "databases": [],
38 "operating_systems": [],
39 "tools": [
40     {
41         "name_tool": "Spark",
42         "level_tool": "Very Good",
43         "duration_tool": null,
44         "is_required": true
45     },
46     {
47         "name_tool": "Azure Databricks",
48         "level_tool": "Good",
49         "duration_tool": null,
50         "is_required": true
51     },
52     {
53         "name_tool": "Airflow",
54         "level_tool": "Good",
55         "duration_tool": null,
56         "is_required": true
57     },
58     {
59         "name_tool": "MLflow",
60         "level_tool": "Good",
61         "duration_tool": null,
62         "is_required": true
63     },
64     {
65         "name_tool": "MLeap",
```

```
66         "level_tool": "Good",
67         "duration_tool": null,
68         "is_required": true
69     },
70     {
71         "name_tool": "Kafka",
72         "level_tool": "Good",
73         "duration_tool": null,
74         "is_required": true
75     },
76     {
77         "name_tool": "Delta Lake",
78         "level_tool": "Good",
79         "duration_tool": null,
80         "is_required": true
81     },
82     {
83         "name_tool": "Azure ADLS",
84         "level_tool": "Good",
85         "duration_tool": null,
86         "is_required": true
87     },
88     {
89         "name_tool": "Azure Devops",
90         "level_tool": "Good",
91         "duration_tool": null,
92         "is_required": true
93     },
94     {
95         "name_tool": "Github",
96         "level_tool": "Very Good",
97         "duration_tool": null,
98         "is_required": true
99     },
100    {
101        "name_tool": "Jira",
102        "level_tool": "Good",
103        "duration_tool": null,
```

```
104         "is_required": true
105     }
106 ]
107 }
108 }
```

Listing C.1: JSON file corresponding to the sample job offer

# D

---

## JSON FILE CORRESPONDING TO THE SAMPLE CV

---

This appendix shows the JSON file corresponding to the sample CV presented in Appendix B.

```
1 {
2   "personal_information": {
3     "name": "Thomas Dixon",
4     "job": "Machine Learning Engineer",
5     "age": "30",
6     "gender": "Male"
7   },
8   "executive_summary": "The consultant is taking a PhD in Computer Vision in
University of Coimbra and hold a master's degree in computer science and
Business Management from ISCTE and a bachelor's degree in Computer Science
and Engineering from Instituto Polytechnic of Castelo Branco. The
consultant started the career in 2014 at IT Sector (Millenium BCP project)
as a Java developer. In 2016, changed to GS1, to work with Outsystems, as
fullstack developer. Between 2017 and 2019, assumed functions as Data
Scientist and Machine Learning Engineer. The consultant was responsible for
developing a Document classification system, where was able to obtain 70%
of accuracy on a multi-label classification problem using TF-IDF and SVM
and worked on an object detection module for automated document parsing (
PDFs) using state of the art Deep Learning techniques. At Celfocus as a
Machine Learning Engineer, was responsible for building and developing
multi label classification models, building machine learning algorithms,
creation of pipeline and a classification model for anomaly detection in
Set up Boxes, using Isolation Forest. Regularly works with Graphs to solve
problems like costumer churn and client segmentation. Currently, works at
Syone managing and leading a team of machine learning composed by 5 members
, present the work done to stakeholders other than that he is working on a
retail client where was responsible to maintain and create new machine
```



```

learning pipelines. The consultant gives python training and architecture
new proposals. The consultant is a team-player and teammate, a good
communicator, that enjoys apply to projects that everyone is committed
working towards the same goal.",
9   "professional_experience": [
10     {
11       "date_from_professional": "2021-05-01",
12       "date_to_professional": "present",
13       "company_name": "Syone",
14       "occupation": "Lead Machine learning Engineer",
15       "sector": "Consultancy",
16       "activities": "During this experience, the consultant was
responsible for the following activities: \n Responsible for leading a team
with 5 members on 2 different projects: customer churn in
telecommunications and building an NLP module that classify sentiment and
extract Entities from phrases;\n Responsible for Cognitive Board where
specific tasks are assigned to each member of the team\n Responsible for
the architecture of the previously modules and designing a REST API to
interact with the modules and integrate with other services;\n Responsible
for mentoring two team members during an internship where was created a
scrapping and classification algorithms. The purpose was to create an
algorithm that gives the best cv match against job offers;\n Responsible to
maintain and add multiple machine learning models in production (
Databricks);\n Development of NLP model to predict whether a review need to
answer or not;",
17       "software_used": "Python, Databricks, MLFlow, Azure Service Studio,
FastAPI Jupyter notebooks, Jira, Gitlab."
18     },
19     {
20       "date_from_professional": "2019-10-01",
21       "date_to_professional": "2021-05-31",
22       "company_name": "Celfocus",
23       "occupation": "Machine learning Engineer",
24       "sector": "Telecommunications",
25       "activities": "During this experience, the consultant was
responsible for the following activities: \n Responsible for building and
developing multi label classification models to classify user requests.
with multi hierarchical levels, three levels. The first level had 17

```

```

categories, we achieve a F-score of 78% on testing set. On the second level
we achieve a F-Score of 60%. We use, TF-IDF, SVM, Naive Bayes, Random
Forest, and KNN to create our models;\n Responsible for building
unsupervised machine learning algorithms (Standard Deviation from average,
MAD, Z-score, Grubbs), and the creation of pipeline for anomaly detection;\n
n Responsible for mentoring one team member in a time series data science
project where we tested unsupervised models MAD, ARIMA, Isolation Forest;\n
Responsible for creating new business value proposals and presenting them
to stakeholders;\n Development of classification model to detect anomalies
in in Set up Boxes using Isolation Forest.",
26     "software_used": "Python, Scala, Apache Spark, Jira, Pyspark, AWS (
S3, EMR), GitLab, Zeppelin, Jupyter Notebooks, Python Bokeh."
27     },
28     {
29         "date_from_professional": "2019-09-01",
30         "date_to_professional": "2020-02-29",
31         "company_name": "ISCTE-IUL",
32         "occupation": "Instructor",
33         "sector": "Teaching",
34         "activities": "During this experience, the consultant was
responsible for the following activities: \n Responsible for create lecture
contents and teaching hands-on classes on Big Data Algorithms.",
35         "software_used": "Apache Spark"
36     },
37     {
38         "date_from_professional": "2017-05-01",
39         "date_to_professional": "2019-10-31",
40         "company_name": "Novabase",
41         "occupation": "Data Scientist and Machine Learning Engineer",
42         "sector": "Finance",
43         "activities": "During this experience, the consultant was
responsible for the following activities: \n Responsible for proposal
creation, status report, in a python module for categorize, project
management documents using Heuristics (specific keywords) and machine
learning techniques (Data Extraction, Data cleaning, TF-IDF, SVM, RF, DT);\n
n Mentoring in a 1-month internship in a data science project whereas the
main purpose is evaluating if the house prices will be increase or decrease
;\n Create a parser library to convert PDF, Image, HTML files into Data

```

```

Objects, Using Heuristics (Regex, Keywords) and Deep Learning for object
detection localization using Faster R-CNN, SSD, YOLO networks;\n Bug fixing
and code refactor on a document platform, which classify documents using
machine learning techniques NLP.",
44     "software_used": "Java, Python, Deep Learning, Image Processing,
Tensorflow."
45     },
46     {
47         "date_from_professional": "2016-04-01",
48         "date_to_professional": "2017-05-31",
49         "company_name": "GS1 Portugal",
50         "occupation": "Full stack developer",
51         "sector": "Retail",
52         "activities": "During this experience, the consultant was
responsible for the following activities: \n Developed a tracking platform
system for food products that come from different countries and are mixed.
e.g.: sausage production.",
53         "software_used": "Outsystems, .Net, JavaScript, CSS, HTML5."
54     },
55     {
56         "date_from_professional": "2015-04-01",
57         "date_to_professional": "2016-04-30",
58         "company_name": "KCSIT - Timwe Group",
59         "occupation": "Backend developer",
60         "sector": "Telecommunications",
61         "activities": "During this experience, the consultant was
responsible for the following activities: \n Developing Billing Api's
integrated with Mobile Operators in order to charge clients when they by
some content on our platform.",
62         "software_used": "Java, Shell Scripting, REST, SOAP, SVN, JUnit,
Tomcat Junit"
63     },
64     {
65         "date_from_professional": "2014-07-01",
66         "date_to_professional": "2015-04-30",
67         "company_name": "IT Sector - Millenium BCP",
68         "occupation": "Backend developer",
69         "sector": "Finance",

```

```

70     "activities": "During this experience, the consultant was
    responsible for the following activities: \n Incident Analysis in Mobile
    and Web Channels of the bank;\n Developing new features and web services
    for web and mobile consumption. Ongoing maintenance.",
71     "software_used": "Java, Shell Scripting, SOAP, JUnit, TFS."
72   }
73 ],
74 "education": [
75   {
76     "date_from_education": "2021-01-01",
77     "date_to_education": "present",
78     "organisation": "University of Coimbra",
79     "subjects": "",
80     "qualification": "PhD in Eletrotechnical Engineering -
    Specialization in Computer Vision"
81   },
82   {
83     "date_from_education": "2017-01-01",
84     "date_to_education": "2020-12-31",
85     "organisation": "ISCTE-IUL - University of Lisbon",
86     "subjects": "",
87     "qualification": "Master's Degree in Computer Science and Business
    management"
88   },
89   {
90     "date_from_education": "2010-01-01",
91     "date_to_education": "2014-12-31",
92     "organisation": "IPCB Institute Polytechnic of Castelo Branco",
93     "subjects": "",
94     "qualification": "B.S in Computer Science and Engineering"
95   }
96 ],
97 "languages": [
98   {
99     "language": "Portuguese",
100    "reading": "Native",
101    "writing": "Native",
102    "speaking": "Native"

```

```
103     },
104     {
105         "language": "English",
106         "reading": "Very Good",
107         "writing": "Very Good",
108         "speaking": "Very Good"
109     },
110     {
111         "language": "Spanish",
112         "reading": "Basic",
113         "writing": "Basic",
114         "speaking": "Basic"
115     },
116     {
117         "language": "German",
118         "reading": "Basic",
119         "writing": "Basic",
120         "speaking": ""
121     }
122 ],
123 "technologies": {
124     "programming_languages": [
125         {
126             "name_programming": "Java",
127             "level_programming": "Good",
128             "date_from_programming": "2014-01-01",
129             "date_to_programming": "2016-12-31",
130             "description_programming": ""
131         },
132         {
133             "name_programming": "Python",
134             "level_programming": "Good",
135             "date_from_programming": "2017-01-01",
136             "date_to_programming": "present",
137             "description_programming": ""
138         },
139         {
140             "name_programming": "Scala",
```

```
141         "level_programming": "Average",
142         "date_from_programming": "2021-01-01",
143         "date_to_programming": "2021-12-31",
144         "description_programming": ""
145     },
146     {
147         "name_programming": "Javascript",
148         "level_programming": "Basic",
149         "date_from_programming": "2017-01-01",
150         "date_to_programming": "2017-12-31",
151         "description_programming": ""
152     },
153     {
154         "name_programming": ".NET",
155         "level_programming": "Basic",
156         "date_from_programming": "2016-01-01",
157         "date_to_programming": "2017-12-31",
158         "description_programming": ""
159     },
160     {
161         "name_programming": "Outsystems",
162         "level_programming": "Average",
163         "date_from_programming": "2016-01-01",
164         "date_to_programming": "2017-12-31",
165         "description_programming": ""
166     },
167     {
168         "name_programming": "HTML",
169         "level_programming": "Average",
170         "date_from_programming": "2016-01-01",
171         "date_to_programming": "2017-12-31",
172         "description_programming": ""
173     },
174     {
175         "name_programming": "CSS",
176         "level_programming": "Average",
177         "date_from_programming": "2016-01-01",
178         "date_to_programming": "2017-12-31",
```

```

179         "description_programming": ""
180     }
181 ],
182 "databases": [
183     {
184         "name_database": "PostgreSQL",
185         "level_database": "Good",
186         "date_from_database": "2018-01-01",
187         "date_to_database": "2019-12-31",
188         "description_database": ""
189     },
190     {
191         "name_database": "Oracle",
192         "level_database": "Good",
193         "date_from_database": "2014-01-01",
194         "date_to_database": "2016-12-31",
195         "description_database": ""
196     },
197     {
198         "name_database": "MSSQL",
199         "level_database": "Average",
200         "date_from_database": "2021-01-01",
201         "date_to_database": "2021-12-31",
202         "description_database": ""
203     },
204     {
205         "name_database": "Big data databases",
206         "level_database": "Average",
207         "date_from_database": "2018-01-01",
208         "date_to_database": "present",
209         "description_database": ""
210     }
211 ],
212 "operating_systems": [
213     {
214         "name_osys": "Linux (Ubuntu, CentOS)",
215         "level_osys": "Good",
216         "date_from_osys": "",

```

```
217         "date_to_osys": "",
218         "description_osys": ""
219     },
220     {
221         "name_osys": "Windows",
222         "level_osys": "Very Good",
223         "date_from_osys": "",
224         "date_to_osys": "",
225         "description_osys": ""
226     }
227 ],
228 "tools": [
229     {
230         "name_tool": "Intellij Idea",
231         "level_tool": "Good",
232         "date_from_tool": "",
233         "date_to_tool": "",
234         "description_tool": ""
235     },
236     {
237         "name_tool": "VS code",
238         "level_tool": "Good",
239         "date_from_tool": "",
240         "date_to_tool": "",
241         "description_tool": ""
242     }
243 ]
244 }
245 }
```

Listing D.1: JSON file corresponding to the sample CV



---

## BIBLIOGRAPHY

---

- Academic-careers-cvs-2017.pdf. <https://cdn.uconnectlabs.com/wp-content/uploads/sites/25/2016/06/Academic-Careers-CVs-2017.pdf>. Accessed: 2021-11-30.
- Caa guidelines | standards & guidelines | caa. <https://www.collegeart.org/standards-and-guidelines/guidelines/visual-art-cv>. Accessed: 2021-11-30.
- Ciência vitae. <https://cienciavitae.pt/>. Accessed: 2021-11-15.
- Home | europass. <https://europa.eu/europass/pt>. Accessed: 2021-11-15.
- Plataforma lattes. <https://lattes.cnpq.br/>. Accessed: 2021-11-15.
- Linkedin portugal: entre ou cadastre-se. <https://pt.linkedin.com/>. Accessed: 2021-11-22.
- Ontodl+. [https://epl.di.uminho.pt/~gepl/GEPL\\_DS/OntoDL/index.html](https://epl.di.uminho.pt/~gepl/GEPL_DS/OntoDL/index.html). Accessed: 2021-11-24.
- Penn treebank ii tag set | clips. <https://web.archive.org/web/20190206204307/https://www.clips.uantwerpen.be/pages/mbsp-tags>. Accessed: 2021-12-14.
- Pretrained models — sentence-transformers documentation. [https://www.sbert.net/docs/pretrained\\_models.html](https://www.sbert.net/docs/pretrained_models.html). Accessed: 2022-06-03.
- Sentencetransformers documentation — sentence-transformers documentation. <https://www.sbert.net/>. Accessed: 2022-05-23.
- Tensorflow hub. <https://tfhub.dev/google/universal-sentence-encoder/4>, a. Accessed: 2022-05-25.
- Tensorflow hub. <https://www.tensorflow.org/hub>, b. Accessed: 2022-05-25.
- Universal pos tags. <https://universaldependencies.org/u/pos/>. Accessed: 2021-12-20.
- sentence-transformers/all-mpnet-base-v2 · hugging face. <https://huggingface.co/sentence-transformers/all-mpnet-base-v2>, a. Accessed: 2022-06-03.
- sentence-transformers/all-minilm-l6-v2 · hugging face. <https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2>, b. Accessed: 2022-06-07.

- Amazon scraps secret ai recruiting tool that showed bias against women. <https://www.reuters.com/article/us-amazon-com-jobs-automation-insight-idUSKCN1MK08G>. Accessed: 2021-11-09.
- sentence-transformers/bert-base-nli-mean-tokens · hugging face. <https://huggingface.co/sentence-transformers/bert-base-nli-mean-tokens>. Accessed: 2022-05-23.
- Chebyshev distance. [https://en.wikipedia.org/wiki/Chebyshev\\_distance](https://en.wikipedia.org/wiki/Chebyshev_distance). Accessed: 2022-07-12.
- Cosine similarity. [https://en.wikipedia.org/wiki/Cosine\\_similarity](https://en.wikipedia.org/wiki/Cosine_similarity). Accessed: 2022-07-11.
- dateparser — python parser for human readable dates - dateparser 1.1.0 documentation. <https://dateparser.readthedocs.io/en/latest/>, a. Accessed: 2022-04-08.
- datetime — basic date and time types - python 3.10.7 documentation. <https://docs.python.org/3/library/datetime.html>, b. Accessed: 2022-04-07.
- dateutil - powerful extensions to datetime — dateutil 2.8.2 documentation. <https://dateutil.readthedocs.io/en/stable/>, c. Accessed: 2022-08-10.
- 4 distance measures for machine learning. <https://machinelearningmastery.com/distance-measures-for-machine-learning/>. Accessed: 2022-07-12.
- re — regular expression operations — python 3.10.7 documentation. <https://docs.python.org/3/library/re.html>. Accessed: 2022-03-03.
- scikit-learn: machine learning in python — scikit-learn 1.1.1 documentation. <https://scikit-learn.org/stable/>. Accessed: 2022-05-19.
- spacy · industrial-strength natural language processing in python. <https://spacy.io/>. Accessed: 2021-12-14.
- Github - chrismattmann/tika-python: Tika-python is a python binding to the apache tika™ rest services allowing tika to be called natively in the python community. <https://github.com/chrismattmann/tika-python>. Accessed: 2022-03-01.
- Calculate distance between two vectors of different length - stack overflow. <https://stackoverflow.com/questions/9314576/calculate-distance-between-two-vectors-of-different-length>. Accessed: 2022-07-12.
- Sujit Amin, Nikita Jayakar, Sonia Sunny, Pheba Babu, M. Kiruthika, and Ambarish Gurjar. Web application for screening resume. In *2019 International Conference on Nascent Technologies in Engineering (ICNTE)*, page 1, 2019.

- Jacob Bank and Benjamin Cole. Calculating the jaccard similarity coefficient with map reduce for entity pairs in wikipedia. In *Wikipedia Similarity Team*, page 5, 2008.
- Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly Media, Inc., 2009.
- Rose Catherine, Karthik Visweswariah, Vijil Chenthamarakshan, and Nanda Kambhatla. Prospect: A system for screening candidates for recruitment. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, page 2, 2010.
- Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Universal sentence encoder. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 1–7, 2018.
- Yongsun Choi, Minh Duc Nguyen, and Thomas N. Kerr. Syntactic and semantic information extraction from npp procedures utilizing natural language processing integrated with rules. In *Nuclear Engineering and Technology*, pages 2, 4, 2021.
- Chirag Daryani, Gurneet Chhabra, Harsh Patel, Indrajeet Chhabra, and Ruchi Patel. An automated resume screening system using natural language processing and similarity. In *Ethics and Information Technology*, page 1, 2020.
- Mihaela-Irina Enachescu. Screening the candidates in it field based on semantic web technologies: Automatic extraction of technical competencies from unstructured resumes. In *Informatica Economica*, page 14, 2019.
- Evandro Fonseca and Joao Paulo Reis Alvarenga. Wide and deep transformers applied to semantic relatedness and textual entailment. In *Oliveira et al.[22]*, pages 1, 5, 2020.
- J. Han, J. Pei, and M. Kamber. *Data Mining: Concepts and Techniques*. Elsevier Science, 2011.
- N. Hardeniya, J. Perkins, D. Chopra, N. Joshi, and I. Mathur. *Natural Language Processing: Python and NLTK*. Packt Publishing, 2016.
- Li-Ping Jing, Hou-Kuan Huang, and Hong-Bo Shi. Improved feature selection approach tfidf in text mining. In *Proceedings. International Conference on Machine Learning and Cybernetics*, page 1, 2002.
- Sven Laumer and Andreas Eckhardt. Help to find the needle in a haystack: Integrating recommender systems in an it supported staff recruitment system. In *Proceedings of the Special Interest Group on Management Information System's 47th Annual Conference on Computer Personnel Research*, page 5, 2009.

- Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning*, pages 3–4, 2014.
- Chew-Hung Lee, Hian Lee, Gee-Wah Ng, and Kheeyin How. Plan ontology and its applications. In *7th Int. Conference on Information Fusion*, page 3, 2004.
- Sameep Mehta, Rakesh Pimplikar, Lav Varshney, and Karthik Visweswariah. Efficient multifaceted screening of job applicants. In *ACM International Conference Proceeding Series*, page 1, 2013.
- Irene Rodrigues, rui rodrigues, and paula couto. Ipr: The semantic textual similarity and recognizing textual entailment systems. In *CEUR Workshop Proceedings*, page 1, 2019.
- José Santos, Ana Oliveira Alves, and Hugo Gonçalo Oliveira. Asapppy: a python framework for portuguese sts. In *ASSIN@STIL*, page 2, 2019.
- Sunita Sarawagi. Information extraction. In *Foundations and Trends in Databases*, pages 9–11, 16, 2008.
- João Souza, Lucas Oliveira, Yohan Gumiel, Deborah Carvalho, and Claudia Moro. Incorporating multiple feature groups to a siamese neural network for semantic textual similarity task in portuguese texts. In *ASSIN@STIL*, page 8, 2019.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 1–15, 2017.