UNIVERSITY OF TWENTE

Identify and extract entities from bibliography references in a free text

MASTER THESIS

Author:
Mattia CHENET

Supervisors:
Dr. Ir. Rieks OP DEN AKKER
Prof. Dr. Dirk HEYLEN

Advisors:
Zubair AFZAL
Dr. Marius DOORNENBAL

A thesis submitted in fulfilment of the requirements for the degree of Human Computer Interaction and Design

in the

Human Media Interaction
University of Twente

September 7, 2017

University of Twente

Abstract

Faculty of electrical engineering, mathematics and computer science
University of Twente

Human Computer Interaction and Design

Identify and extract entities from bibliography references in a free text

by Mattia CHENET

Elsevier is the world's largest scientific publishing company. It owns a database named Scopus that stores a multitude of scientific papers, books and manuscripts. Besides scientific documents, author's profiles are also stored. Author's profiles contain information such as documents published and the number of citations that an author gets on his works from other scientific publication. Sometimes scientific articles are missing or the number of citation can be lower than the author expected. There are several reasons why a scientific report is missing in the author profile or the number of a citation can be lower than expected. One of those is because the document can be out of policy and therefore not yet referenced in the database. In such cases the author can contact the Scopus customer service by email including the reference of the missing scientific document. These references are written within the text of the email using several styles and are often incomplete. Sometimes the year of publication is missing, only the first author is mentioned, or the title is not complete.

Elsevier is developing a technology that aims to support the Customer Service operator to a faster understanding of the problem behind a missing scientific document record. In order to understand and automate those problems, such as automatic recognition out of policy papers, the machine has to be able to understand which documents are referenced within the text of the emails. In order to extract the right entities and retrieve the correct document mentioned in the free text, such as emails or web forms, this research project has been divided into three main steps. The first step is to identify the parts of a text that refer to a bibliographic referent, the reference text. The second step is to identify the components of a reference text or references' entities, e.g. author name(s), title, year of publication, journal name, etc. The third step is to find a match with the referent in the text and the real document according to the components identified.

All of the above have led to the following question: can each of these steps be done fully automatically using machine learning approaches in order to retrieve the correct scientific document included by the reference in the email?

This report presents the answer to this question. It focuses on references' entities recognition in emails data, considered free text. It has been found that narrowing down the problem by dividing and classifying the email's sentences that contain the references and subsequently performing entities extraction on these sentences, it is possible to obtain a good performance in terms of the reference entities extraction. By having good accuracy on entities recognition it is possible to recognize and retrieve the corrected scientific document mentioned within an unstructured (free) text.

Acknowledgements

In order to carry out this research project, many people help me, therefore I want to thank them.
At first, my advisor at Elsevier, Afzal Zubair that follow me during the whole experiment. He is patient and capable to teach me how to develop an NLP software, he made me thinking about each step to take it by sharing his knowledge and experience with me, beyond a supervisor is also an amazing teacher.
I want to thanks, my University Supervisors Rieks Op Den Akker and Dirk Heylen that first helping me contact Elsevier for this interesting project and then follow and help me to carry out this final report by supervising the process and the work done here in Elsevier.
I want to thank Marius Doornenbal, my boss and advisor in Elsevier to gave me this opportunity to be part and work with amazing people in the Content and Innovation Department.
Umesh, Zhemin, Saber, Ricardo, Thimeos, Yi, Arthur, Alexandros and in particular Sergios Petridis, Kayal Deep and Eustratiadis Panagiotis, the NLP specialists that besides enjoy together everyday morning stand up meetings, they also help me with advice on action to take during the development process of this experiment.
Last but not least my desk-mate Marialaura, she took care of me during this 6 month of internship.

Contents

List of Figures

List of Tables

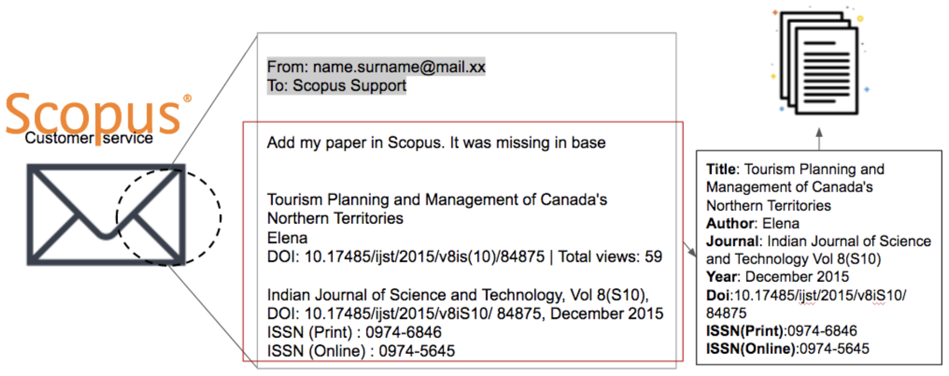
Dedicated to Mum, Dad, and Brother.
To the many friends that I met on my way during those 2 years of abroad studies.

Chapter 1

Introduction

Elsevier is a Scientific publications leading company established in 1880. Every year it publishes more than 420,000 scientific articles, book and manuscripts. Besides publishing and printing journals and articles Elsevier has several products that position it as the leading information provider for the scientific and corporate research community. One of these products is Scopus.
Scopus is the largest abstract citation database of peer-review literature. It stores information about scientific documents and authors. For each author it provides information such as published documents or number of citations that an author got from other scientific publications.
Sometimes, in the author's profile of Scopus, it occurs that not all published articles are referenced or the number of citations is lower than expected. This can happen for several reasons, for example, the article might still be in the production phase, hence, not already published. Alternatively the article might be out of policy document, hence, not referenced on Scopus database. In this case the author might contact the Scopus customer service by email, wondering why his profile is not up to date. Inside the email's text, the author will include the reference to the document that is missing in his profile. These references, written in the email text, do not follow any style and are often incomplete.
Sometimes entities such as year of publication is missing, only the first author is mentioned, or the title is not complete.
Because of this problem, Elsevier is in the process of developing technology that will help the Scopus customer service to automatically identify a problem behind an out of policy or still in production document and consequentially reducing its requests backlog.
In order to automatize and solve these problems, at first, the machine has to understand which document is contained in informal text or unstructured text such as emails.
In other words, we have to be able to process unstructured(or free) text in order to understand to which document is referring the author in the email.
Therefore this experiment is decomposed in three main step.
The first step, in this matching process, will identify those parts of the text that refers to a bibliographic referent, the reference text. The second step is to identify the entities that compose the reference text, such as author name(s), title, year of publication, journal name, and other entities when they exist.
The third step is to normalize the data extracted and match them with the referent document.
We trained and used several machine learners and pre-trained algorithm, tested their performances on references recognition and later entities extraction.
In the following first chapter, an introduction to references, structure and unstructured text is given. Machine learning and natural language processing are briefly

introduced. This chapter concludes with the research questions addressed throughout this experiment, the challenges met and the contribution that this experiment can bring to Scopus customer service and can lead to the development of other automated systems.



From: name.surname@mail.xx
To: Scopus Support

Add my paper in Scopus. It was missing in base

Tourism Planning and Management of Canada's Northern Territories
Elena
DOI: 10.17485/ijst/2015/v8is(10)/84875 | Total views: 59

Indian Journal of Science and Technology, Vol 8(S10), DOI: 10.17485/ijst/2015/v8iS10/ 84875, December 2015
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Title: Tourism Planning and Management of Canada's Northern Territories
Author: Elena
Journal: Indian Journal of Science and Technology Vol 8(S10)
Year: December 2015
Doi:10.17485/ijst/2015/v8iS10/ 84875
ISSN(Print):0974-6846
ISSN(Online):0974-5645

FIGURE 1.1: Goal of the experiment

1.1 What is a reference

The scientific community uses certain pre-defined rules to represent a reference or a citation within the text of scientific article, book or publication:

1. In-text references or in-text citation when you refer to, summarize, paraphrase, or quote form another source. For every in-text citation refer to a list of references at the end of the document called “bibliography”.
2. A Bibliography is by definition a list of source materials that are used or consulted in the preparation of a work or that are referred to in the text. The list is composed by references or items [11].
3. Bibliography items or bibliography references is the actual piece of text where the information about the source is given.

1.2 Structured text and unstructured text

1.2.1 Structured text

By "structured text" is refer to text that has a predefined structure and it follows certain conventions. A reference collects information about the source mentioned in the document on both printed and web site sources. The components or entities of a reference change according to the source, if is referring either to a printed document or a web site.
In general, a reference comprehend the entities as follow:

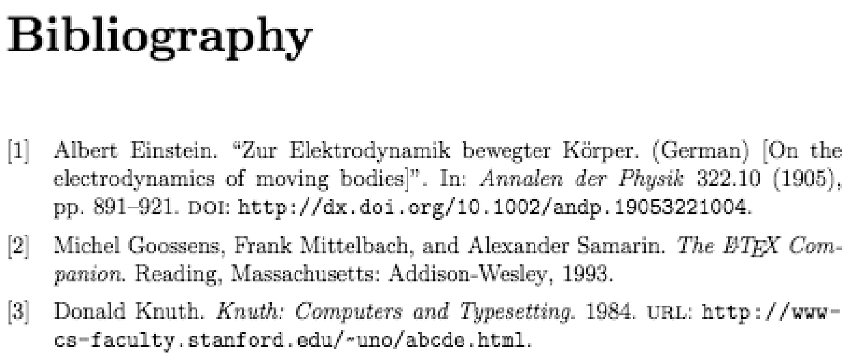
- author name
- title of the publication (and the title of the article if it’s a magazine or encyclopedia)
- date of publication
- the place of publication of a book

- the publishing company of a book
- the volume number of a magazine or printed encyclopedia the page number(s)

The information about the Web sites source follow this rules:

- author and editor names (if available)
- title of the page (if available)
- the company or organization who posted the webpage
- the Web address for the page (called a URL)
- the last date you looked at the page

When the author collects all the information about the sources, he will put them in a list at the end of the document. This list, also known as bibliography, contains all the source's references and for each reference the relative entities mentioned briefly above.
Each reference's entity is mentioned in the Bibliography according to the style that an author chooses. There are several styles that authors can choose to describe the bibliography[26], APA style (American Psychology Association) and MLA style (Modern Language Associate) are the two styles broadly used for describing bibliographies. Below these styles are explained in details.


Bibliography

[1] Albert Einstein. "Zur Elektrodynamik bewegter Körper. (German) [On the electrodynamics of moving bodies]". In: *Annalen der Physik* 322.10 (1905), pp. 891–921. DOI: `http://dx.doi.org/10.1002/andp.19053221004`.

[2] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*. Reading, Massachusetts: Addison-Wesley, 1993.

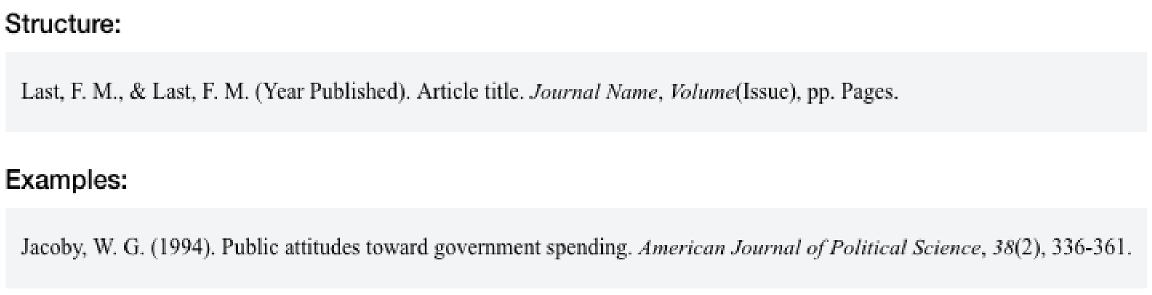
[3] Donald Knuth. *Knuth: Computers and Typesetting*. 1984. URL: `http://www-cs-faculty.stanford.edu/~uno/abcde.html`.

FIGURE 1.2: An example of bibliography with 3 references

APA style

Each style defines the way how a reference should be written. The APA style [4] prescribes that the authors names should be alphabetized by the last name of the first author of each work. If more than one author is cited, each author has to be separate by commas alphabetically, the last name for each author it should be mentioned followed by the initials of the first and the middle name. If there are more than six authors cited after the sixth it will be follow from "et al." ("and others" from Latin language) After the authors it should follow by the publication year, than the title of the source with the source (Journal or book) followed in italic style. The number of the volume followed by the issue in round brackets and the pages (referring by p. if is one pp. if is a range of pages) are the last element in the reference.
Those rules are applied on cited web sites as well. APA for web sites. uses the same style with the authors followed by the date of publishing in brackets, the article title

in italic style and the retrieved Url.



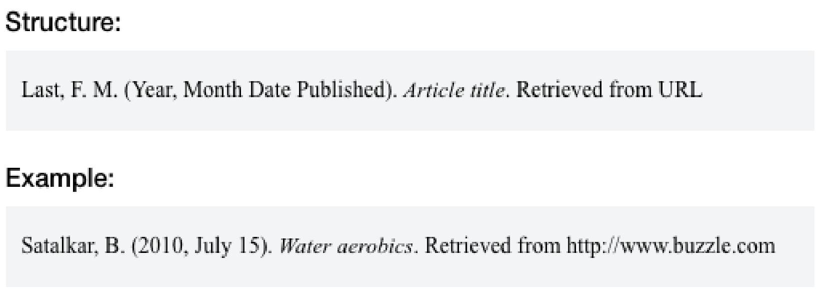
Structure:

Last, F. M., & Last, F. M. (Year Published). Article title. *Journal Name*, *Volume*(Issue), pp. Pages.

Examples:

Jacoby, W. G. (1994). Public attitudes toward government spending. *American Journal of Political Science*, *38*(2), 336-361.

FIGURE 1.3: An example of journal reference APA style



Structure:

Last, F. M. (Year, Month Date Published). *Article title*. Retrieved from URL

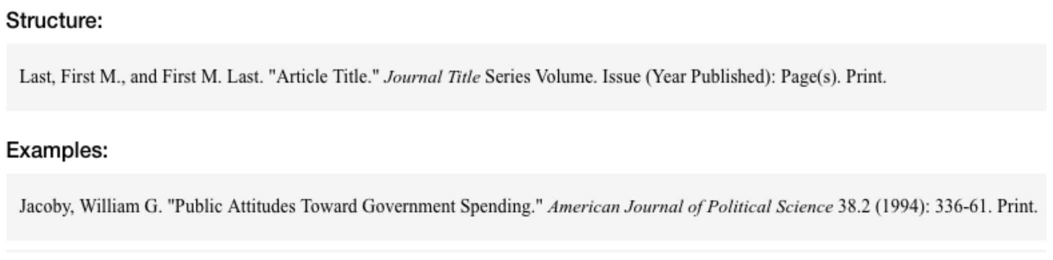
Example:

Satalkar, B. (2010, July 15). *Water aerobics*. Retrieved from http://www.buzzle.com

FIGURE 1.4: An example of web reference APA style

MLA style

MLA style [2] includes the author's name, the title of the article, the name of the journal, the series number/type of the journal (if given), the volume number (if given), the issue number (if given), the year of publication, the page numbers of the article. The author should be written as plain text First and Last name divided by comma followed by the initial middle name. The title is contained in quotes followed by the journal title in italic style the volume and the issue separated by a dot. Finally, the year as round brackets and the ranges of pages of the journal.
Web sites citation use the style for authors and the title as the same of above, followed by the website title in italic style, the publisher of the website and the date of the published article.



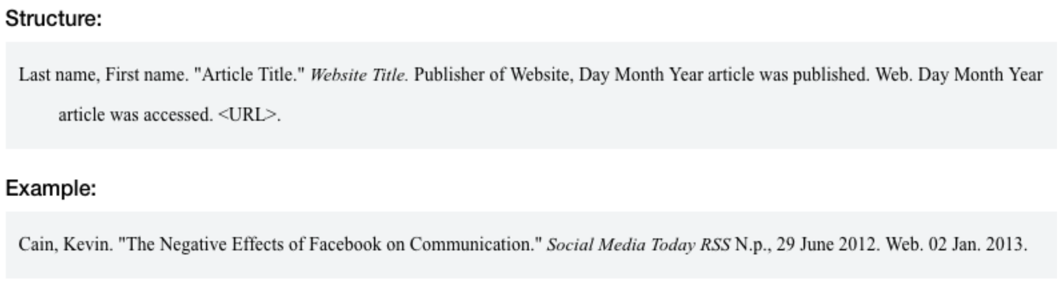
Structure:

Last, First M., and First M. Last. "Article Title." *Journal Title* Series Volume. Issue (Year Published): Page(s). Print.

Examples:

Jacoby, William G. "Public Attitudes Toward Government Spending." *American Journal of Political Science* 38.2 (1994): 336-61. Print.

FIGURE 1.5: An example of journal reference MLA style


Structure:

Last name, First name. "Article Title." *Website Title.* Publisher of Website, Day Month Year article was published. Web. Day Month Year article was accessed. <URL>.

Example:

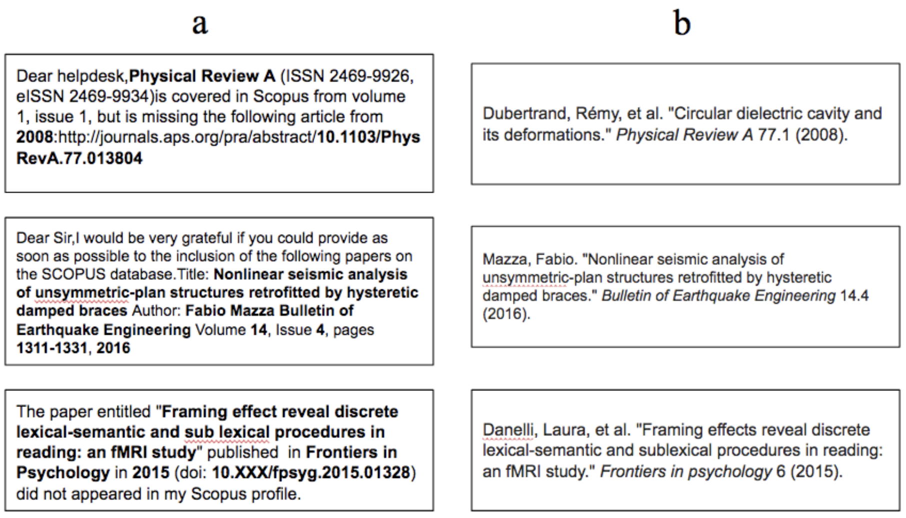
Cain, Kevin. "The Negative Effects of Facebook on Communication." *Social Media Today RSS* N.p., 29 June 2012. Web. 02 Jan. 2013.

FIGURE 1.6: An example of web reference MLA style

Those styles and many others are used for listing the references in the bibliography. The bibliography therefore is considered a structure text because it follows those rules that aim to standardize it for different purposes such as indexing and analytics.

1.2.2 Unstructured text

The unstructured text refers to text that is not standardised or organised in a predefined manner. Is considered unstructured text or free text, the text that can type in an email, web-forms, instant messages or social media posting such as tweets or Facebook post, where the user liberally decides how to structure it and which rules to follow. Mining of unstructured text delivers new insights by uncovering previously unknown information, detecting patterns and trends, and identifying connections between seemingly unrelated pieces of data [32]. Natural language processing software and other automated tools are typically used to prepare the unstructured text in such a way that also machine and systems can understand it and process it. Because the language usually is different and vague is necessary, at first, understand the context is an essential step in the mining process. The content is also reviewed for word frequency and other patterns. Tagging is performed to label various pieces of text-derived data so it can be categorised and grouped in ways that are most likely to deliver useful information. Once the text has converted into data, it can be analysed and evaluated for relevance and importance. Below are shown some examples of an unstructured text and structured text that we face during this research project. In the left squares the unstructured text. In bold are shown the reference's entities that we have to extract from the unstructured text.


a

Dear helpdesk,**Physical Review A** (ISSN 2469-9926, eISSN 2469-9934)is covered in Scopus from volume 1, issue 1, but is missing the following article from **2008**:http://journals.aps.org/pra/abstract/**10.1103/PhysRevA.77.013804**

Dear Sir,I would be very grateful if you could provide as soon as possible to the inclusion of the following papers on the SCOPUS database.Title: **Nonlinear seismic analysis of unsymmetric-plan structures retrofitted by hysteretic damped braces** Author: **Fabio Mazza Bulletin of Earthquake Engineering** Volume **14**, Issue **4**, pages **1311-1331**, **2016**

The paper entitled "**Framing effect reveal discrete lexical-semantic and sub lexical procedures in reading: an fMRI study**" published in **Frontiers in Psychology** in **2015** (doi: **10.XXX/fpsyg.2015.01328**) did not appeared in my Scopus profile.

b

Dubertrand, Rémy, et al. "Circular dielectric cavity and its deformations." *Physical Review A* 77.1 (2008).

Mazza, Fabio. "Nonlinear seismic analysis of unsymmetric-plan structures retrofitted by hysteretic damped braces." *Bulletin of Earthquake Engineering* 14.4 (2016).

Danelli, Laura, et al. "Framing effects reveal discrete lexical-semantic and sublexical procedures in reading: an fMRI study." *Frontiers in psychology* 6 (2015).

FIGURE 1.7: Differences between unstructured text and structure text. A: unstructured text, in bold the entities to extract; B: structure text (MLA style)

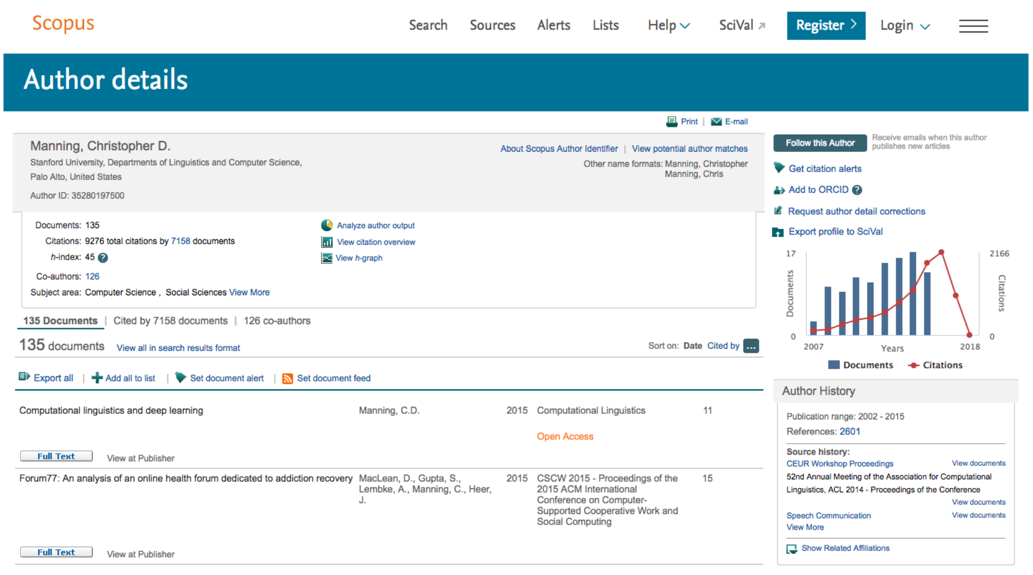
1.3 Introduction to Elsevier and Scopus database

Elsevier publishes approximately 420,000 articles annually in 2,500 journals including impactful journals such as Tetrahedron, Cell and The Lancet. Elsevier not only publishes paper scientific journals and documents but also produce and develop software products such as ScienceDirect, Reaxys and Scopus, digital libraries that give access to millions of scientific documents to companies and researchers.
In this way, Elsevier is positioning itself as the leading information provider for the scientific and corporate research community. Scopus, for example, is the largest abstract and citation database of peer-reviewed literature: scientific journals, books and conference proceedings. Delivering a comprehensive overview of the world's research output in the fields of science, technology, medicine, social sciences, and arts and humanities. Scopus features smart tools to track, analyse and visualize researchers. Scopus database containing abstracts and citations for academic journal articles. It covers nearly 22,000 titles from over 5000 publishers, of which 20,000 are peer-review journals in the scientific, technical, medical and social sciences. It stores more than 35 million authors profile and approximately 70 million scientific documents. In the following session we are going to describe the workflow of Scopus Customer service and how the technology developed in this research project aim to support it.

1.3.1 Scopus costumer service

Scopus is used from more the 3,000 academic, government and corporate institutions.
As briefly mentioned above Scopus store approximately 35 million authors profile. Authors profile stores several types of information, in particular the articles and documents that an author has been published and the amount of citations that each work got from other scientific experiments. Due to all this great number of people that use Scopus and the amount of information that is stored, in the author's profile, occasionally some scientific documents are missing or the number of citation is lower then expected. Scopus, therefore, has a customer service that gives support to

customers (authors) by helping them to understand why a paper is missing or why the number of citation is lower than how it should be. These problems can happen for two reasons: the article that is citing is not listed in Scopus, also known as *out of policy papers*, or the article can be still in production phase and consequentially not fully published.
All the process of correction is made by emails exchange, where authors will include the reference of the document in question. The procedure to verify and correct these kinds of problems is not immediate but, instead is extremely time-consuming for both the customer service operator and the author. Between a correction of a problem can pass weeks or even months depending on several factors. For example, if the requests backlog of the Customer service is higher, the customer service operator will take more time to answer to all of them.
Elsevier is developing technologies that aim to improve the Scopus customer experience by reducing the requests backlog on problems classification , hence, increasing the customer service operator efficiency. In order to automatically identify problems such as out of policy paper or missing citations, at first, the machine has to understand which document the author includes in the email text. Therefore this research project focuses on reference's entities recognition and extraction. By extracting the reference's entities with high accuracy we are able to retrieve the correct scientific document mentioned in the email. Just in 2016, the Scopus customer service received more than 74,000 emails, more than 200 emails per day. This final project takes inspiration from this amount of data that every day the customer service received from their customer. Furthermore, when the machine will be able to automatically identify the scientific document in the unstructured text it will be possible to develop and automate the resolution of problems such as identity *out of policy papers* or documents that still in the production phase, without using any Scopus customer service operator efforts, hence decreasing the Scopus customer service requests backlog.



Author details

Print | E-mail

Manning, Christopher D.
Stanford University, Departments of Linguistics and Computer Science, Palo Alto, United States
Author ID: 35280197500

About Scopus Author Identifier | View potential author matches
Other name formats: Manning, Christopher
Manning, Chris

Documents: 135
Citations: 9276 total citations by 7158 documents
h-index: 45
Co-authors: 126
Subject area: Computer Science , Social Sciences View More

135 Documents | Cited by 7158 documents | 126 co-authors

135 documents View all in search results format Sort on: Date Cited by

Export all | Add all to list | Set document alert | Set document feed

Computational linguistics and deep learning	Manning, C.D.	2015	Computational Linguistics Open Access	11
Full Text View at Publisher				
Forum77: An analysis of an online health forum dedicated to addiction recovery	MacLean, D., Gupta, S., Lembke, A., Manning, C., Heer, J.	2015	CSCW 2015 - Proceedings of the 2015 ACM International Conference on Computer-Supported Cooperative Work and Social Computing	15
Full Text View at Publisher				

Follow this Author Receive emails when this author publishes new articles

Author History
Publication range: 2002 - 2015
References: 2601
Source history:
CEUR Workshop Proceedings View documents
52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014 - Proceedings of the Conference View documents
Speech Communication View documents
View More
Show Related Affiliations

FIGURE 1.8: An Example of author profile

1.4 Introduction to machine learning

Machine learning and data mining are applied in a lot of different areas. Machine learning is used, for example, in retail for targeting advertising, in finance banks

analyse their past data to build models to use in credit applications, fraud detection and stock market analysis [6]. In manufacturing, learning models are used for optimisation, control and troubleshooting. In medicine, learning programs are used for medical diagnosis. In telecommunications, call patterns are analysed for network optimisation and maximising the quality of the service. In science, significant amounts of data can only be interpreted fast enough by computers. All the volume of information in the worldwide web is enormous and steadily growing, searching for relevant information cannot be done manually.
Machine learning is part of artificial intelligence. To be intelligent, a system should learn from the changing environment. Therefore, the system can learn and adapt to such changes, so the designer does not need to provide solution to all the possible cases [1].
Our case scenario suits perfectly as a typical machine learning problem. The references and his entities are written in many forms and styles; there are no rules to write the references in emails. We build a system that adapts and recognise the reference and its entities with high accuracy, leveraging machine learning techniques. Classify and extract references and its entities on text is also known as Natural language processing and text mining problem.

1.4.1 Natural language processing and text mining

When it comes to analysing unstructured data sets, a range of methodologies are used. Text mining, often referred as text analytics is by definition a process or practice of examining large collections of written resources to generate new information[23]. In other words, the goal of text mining is to discover relevant information in a text by transforming it into data that can be used for further analysis. The primary goal of text mining is to find relevant information that is possibly unknown and hidden in the context of other information. Text mining accomplishes this by using several methodologies; Natural language processing is one of them. Natural Language processing, also known with the abbreviations NLP, is a component of text mining that uses a variety of methodologies to decipher the ambiguities in human language. NLP performs a particular kind of linguistic analytics such as automatic summarisation, part-of-speech tagging, disambiguation, entity extraction and relation extraction. Text mining and NLP are commonly used together for different purposes, as, one of the most common applications, called sentiment analysis, is applied to social media monitoring, where an analysis is performed on a pool of user-generated content to understand mood, emotions and awareness related to a topic. This experiment will use NLP techniques along with text mining to gain insight and to extract features that will help us to understand the information from unstructured text.

1.5 Research question and challenges

1.5.1 Research question

The main research question is summarized below, together with several subsequent sub questions

Given unstructured text, text that can be fetch in emails, web-forms, instant messages or social media posting where is contain information about references or citations, how can machine learning approaches perform on the task of recognize and extracting entities such as Title, Authors, Journal Title, Volume, Issue, year, pages and others when those are mentioned in the unstructured text.

To carry out some experiments to try answering this question, following two sub-questions are also crucial to consider:

1. Can we identify a snippet in the unstructured text that is consider as containing a reference?
2. How can we identify and extract the entities that belong to the reference contain in the snippet of unstructured text?

1.5.2 Challenges

To answer to the research and the sub-questions listed above, there are several challenges to be addressed.
At first, as mentioned before and shown in the figure 1.7, this research project deal with unstructured text, the text can be written in several ways or styles, therefore is difficult to process with heuristics-based algorithms because the cases are too many. Moreover, there will be some parts of it that are unuseful and others that will contain the information that we want to extract. Therefore, the first challenge will be excluding those parts of the text that are not valuable for the purpose of this experiment. We want to perform entities extraction on the part of the text where the reference is contained. This leads to a challenging task which is how the text will be split for subsequent reference identification. When this portion of text will be defined by tokenizing the text in snippets (or sentences), the second challenge will be, hence, classify each sentence as containing the reference or not.
In order to understand which documents are mentioned in the unstructured text, each sentence predicted as containing a reference it is taken in input for entities extraction. The last challenge will be the output normalization of each entity in order to retrieve the correct paper.

1.6 Contribution

This final project will bring the following contributions :

1. An example of how machine learning approaches can deal with unstructured text, aiming to leads to subsequent solution in order to automate and improve the efficiency of the customer service, such as automatically identity out of policy scientific article or still in production document.
2. How to build a golden-set from raw data. We deal with emails, therefore, we will start building our golden-set starting with raw data by performing annotation and use them in input for training the models.
3. Several approaches for dealing with unstructured text are given. Set of features for identifying references in a free text are proposed. These features can bring insight to the researcher for subsequent experiments.

1.7 Overview

Chapter 2 will explain how information extraction on scientific papers and documents theory is developed and used in different scenarios by other researchers. Subsequently, a discussion on how different methods are applied to handle entities extraction and related tasks is given, including both rules-based approaches and machine learning-based approaches.
Chapter 3 describe the data used for build the golden-set how we select our subset of emails, how and why we annotated those emails and which tools we used.
Our methodology and details of our experiments can be found in Chapter 4, which covers the features involved in our experiments and which kinds of machine learning models we use. In Chapter 5, results of the experiments are explained for both our binary classifiers and ensemble methods, along with the systems that we selected for perform entities extraction evaluation. In Chapter 6 discussions on the results obtained from the experiment are described. This report ends with conclusion and future work of this research in Chapter 7.

Chapter 2

Related Works

2.1 Information extraction from scientific manuscript

Scientific literature is becoming larger day by day. As a consequence, finding relevant information and relevant papers have become more difficult [26]. Therefore scientific community is constantly developing methods and tools to extract information from scientific documents. Their goal is building systems that help scientist to find the right information they are looking for.
Although a scientific article is a structured document, extract the right information is still challenging due to the different style of writing each document section. There are many approaches that focus on extracting information from scientific articles. Scientific articles are structured documents, there are several conventions to follow for writing them correctly. Using machine learning techniques or heuristics for extracting information allows utilizing the metadata extracted for subsequent use for analysis or assist the automatic index creation for digital libraries such as Scopus or Google Scholar. We studied two types of approaches to information extraction in scientific documents, rules based approach that apply heuristics techniques such as regular expression or logic programming and machine learning approaches such as text mining combined with natural language processing.
Most of the works often do not process the entire input document. Some works focus on extracting information from part of the section such as parts of the abstract, which is considered unstructured text as Dasigi, Pradeep et al. did in [10] and Lin et al. in [19].
Zhang, Xiaoli, et al. in [36] and Zou et al. in [37] focus their experiments on the extraction of information from part of the bibliography to classify and extract reference information. Other works focus on extracting more portions of the whole article, especially in PDF files like in CERMINE[34] which is a system based on a modular workflow, who is loosely coupled architecture allows for individual component evaluation and adjustment. Studied solutions are usually based on heuristics and machine learning techniques.
This chapter will go through the studies that were carried out with the aim to get suggestions on the approach to follow for reach the goal of our research experiment on unstructured texts. At first, rules and machine learning studied approaches are explained and some example are given. Lastly, this Chapter, explain in deep how the three selected systems for evaluation on entities extraction work.

2.2 Rules-based approach

Giuffrida et al. in [12] used a rules-based methodology to extract metadata from structured files written in a dynamically typed concatenative programming language called PostScript. PostScript is used in electronic and desktop publishing areas as a page description language [16]. Constantin et al. [7] proposed PDFX. This tool is used for converting a scientific document in PDF format to XML language. It allows extracting metadata from references in the Bibliography and other necessary metadata from full structured texts. Pdf-extract [27] is an open-source tool developed by Crossref lab it identifies, at first, the region of the scientific document and later it extracts the related metadata. It uses a combination of visual cues and content traits to perform structural analysis that determines the regions of PDF documents such as columns, headers, footers and sections, detect references sections and extract individual references with the related entities.

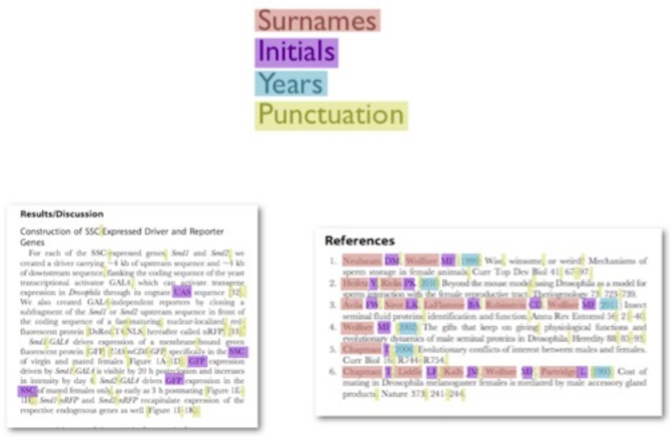


FIGURE 2.1: Rule-based approach on entities extraction from Pdf-extract[27]

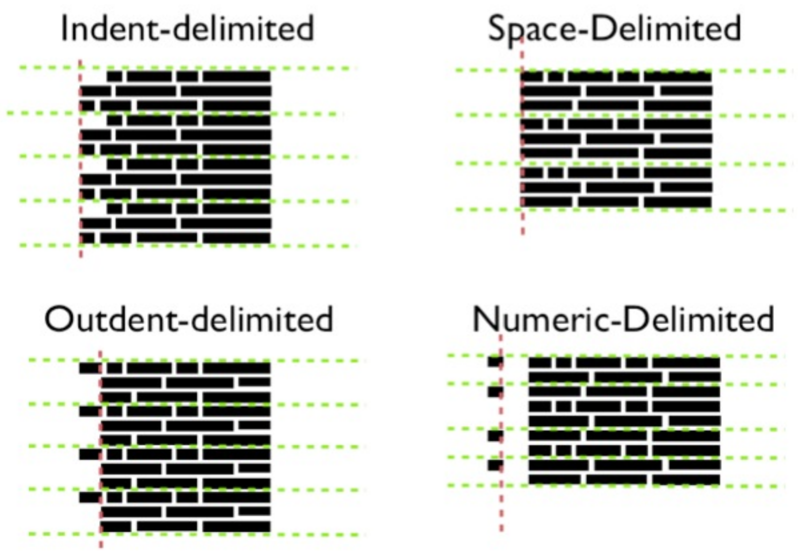


FIGURE 2.2: Rule-based approach on bibliography recognition from Pdf-extract[27]

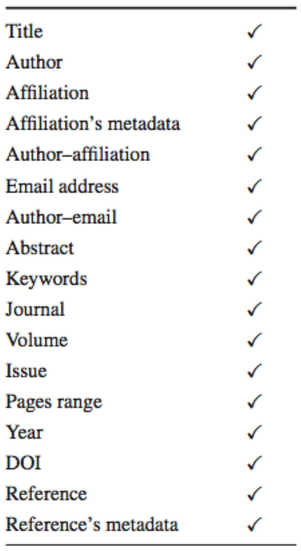
2.3 Machine learning model-based approach

Machine learning approaches are more employed than the rule-based ones when it comes to extracting metadata from scientific documents. They are more popular because they can be applied to several types of structured text thanks to their capabilities of adapting to a changing environment such as different document's structures. Most of them use the same approach that Pdf-extract follows, by firstly dividing the document into regions and later perform metadata extraction according to the region of the document. For example, Han et al. in [14] and CRIS systems proposed by Kovacevic et al. [18] extract metadata from portions of scientific documents in PDF format by applying two-stage classification of text-lines with the use of Support Vector Machines (SVMs) in combination with text-related features. These tools classify at first the region of the document using geometric features and then they extract the related metadata. Lu et al. [21] use a combination of both machine learning algorithm and rule-based approaches in order to obtain volume level, issue level and others articles metadata. In their approach, the pages are first parsed by the OCRed algorithm, a rule-based pattern matching used for volume, issue, title, and range of pages, while article metadata is extracted using SVM and both geometric and textual features extracted from the content. Structural SVM approach was used in [36] by Zhang, Xiaoli et al. They compared structural SVM with conventional SVM observing that structural SVM and Conditional Random Fields (CRF) achieve about the same accuracies at token and chunk levels, showing that structural SVM has more advantages than the traditional ones.
Other experiments use different classification algorithms such as Hidden Markov Models (HMM), neural classifiers, Maximum Entropy and CRF. Marinai [24] extracts metadata from PDF by employing neural classifier for zone classification. Cui and Chen [9] use HMM classifier to extract metadata from PDF scientific documents. Kern et al. [17] can extract a basic set of metadata from PDF documents using an enhanced Maximum Entropy classifier. GROBID proposed by Lopez [20] is a system that analyzes and extracts metadata from scientific texts in PDF format. It uses CRF in order to extract document's metadata, full text and a list of parsed bibliographic references. ParsCit, by Luong et al. [22] uses CRF for extracting the logical structure of scientific documents, including the references from the bibliography. Zou, Jie et al. in [37] compared SVM and CRF approaches for reference parsing from HTML structured files. They first divided the experiment into 2 tasks. First, they locate the reference using LibSVM tool. An SVM classifier was assigned each zone a probability value for being a reference zone. The authors, title and other 5 entities were extracted from the reference by parsing it using two classifiers SVM and CRF for evaluating one method against the other. Lin, Ryan proposed a CRF implementation in [19] to automatically identify the results-conclusions section in biomedical abstracts. They used different sets of features such as NE features, NER numerical normalization & pattern extraction, tense feature and WUF & WBF features (unigram and bigram word frequency). Tkaczyk, Dominika et al. propose CERMINE [34] an open-source system for extracting structured metadata from scientific articles in a born-digital form. CERMINE use several implementations dived in 3 parts a Basic structure recognition, the metadata extraction through an SVM classifier and simple rule-based approaches for the extraction of entities such as article title and author affiliation. Cermine is able to recognize Bibliography and entities extraction using K-means clustering and CRF models. In 2017 Pradeep Dasigi et al. proposed in [10] a deep learning model for identifying structure within experiment narratives in scientific literature. They took this problem as a sequence labeling

approach and they labeled clauses within experiment narratives to identify the different parts in the document. The dataset consists of paragraphs taken from open access PubMed papers labeled with rhetorical information as a result of our pilot annotation. Their model is a Recurrent Neural Network (RNN) with Long Short-Term Memory (LSTM) cells that label clauses. The clause representations are computed by combining word representations using a novel attention mechanism that involves a separate RNN. The model was comparing LSTMs where the input layer has simple or no attention against CRF model showing that LSTMs could reach slightly better performances.

2.3.1 Cermine

As mentioned above Cermine by Dominika Tkaczyk et al.[34] was developed by the researcher of the University of Warsaw in 2015. Cermine from the name Content ExtRactor and MINEr is an open-source Java library and a web service for extracting metadata and content from PDF file comprehending scientific content. The system implementation of most steps is based on supervised and unsupervised machine learning techniques, which permit Cermine to adapt to numerous styles and layout for extract the correct information. The authors wanted the algorithm to extract information from several types of structured documents despite gaining a respectable performance. Cermine can not only extract metadata from PDF scientific article but also from affiliation and references string. It achieved an average F-score of 77.5% on meteadata extraction form several types of structured document. Cermine evaluation was compared with similar systems showing better performance for most of metadata extraction types. Here in the table below are described which type of metadata Cermine recognize and extract from the scientific documents.



Title	✓
Author	✓
Affiliation	✓
Affiliation's metadata	✓
Author–affiliation	✓
Email address	✓
Author–email	✓
Abstract	✓
Keywords	✓
Journal	✓
Volume	✓
Issue	✓
Pages range	✓
Year	✓
DOI	✓
Reference	✓
Reference's metadata	✓

FIGURE 2.3: Cermine entities recognition[34]

Cermine and the bibliography entities extraction tool was selected in our evaluation for his ability to adapt and maintain good performances on the extraction on several styles a layout of strings. It comes with Jar library downloadable from its github repository[1].

[1]https://github.com/CeON/CERMINE

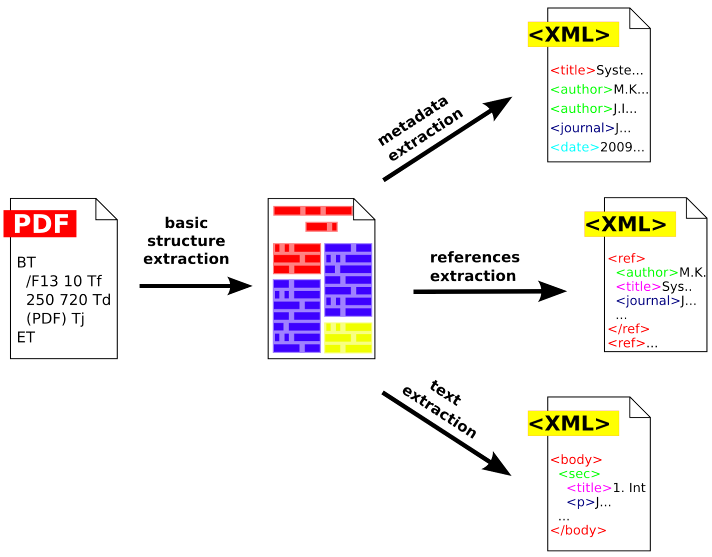
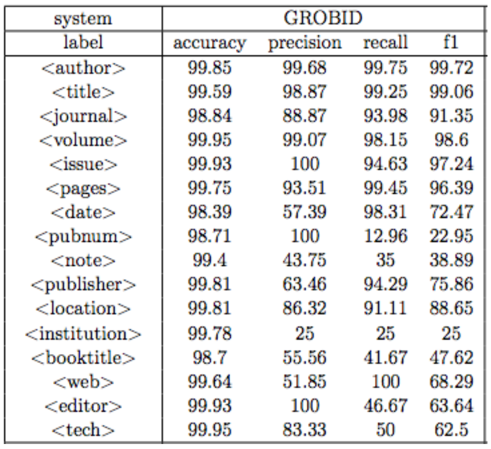


FIGURE 2.4: Work flow of Cermine[34]

2.3.2 Grobid

Grobid developed in 2011 by Patrice Lopez [20] perform bibliographic data extraction from scholar articles combine with a multi-level term extractions. Grobib was one of the first libraries that were developed for the purpose to show the power of machine learning involved in bibliography data extraction in scholarly articles. This tool is one of the standards adopt by digital library community for parsing bibliography references with respectable performances and high accuracy as shown in the table below



system	GROBID			
label	accuracy	precision	recall	f1
<author>	99.85	99.68	99.75	99.72
<title>	99.59	98.87	99.25	99.06
<journal>	98.84	88.87	93.98	91.35
<volume>	99.95	99.07	98.15	98.6
<issue>	99.93	100	94.63	97.24
<pages>	99.75	93.51	99.45	96.39
<date>	98.39	57.39	98.31	72.47
<pubnum>	98.71	100	12.96	22.95
<note>	99.4	43.75	35	38.89
<publisher>	99.81	63.46	94.29	75.86
<location>	99.81	86.32	91.11	88.65
<institution>	99.78	25	25	25
<booktitle>	98.7	55.56	41.67	47.62
<web>	99.64	51.85	100	68.29
<editor>	99.93	100	46.67	63.64
<tech>	99.95	83.33	50	62.5

FIGURE 2.5: Grobid entities recognition form scientific document[20]

The automatic extraction of bibliographical data is a challenging task because, as mentioned in the previous sections, even if scientific documents and bibliographies are considered as a structured text, they present a high variability of the formats. Grobid it comes with several functionalities that perform document's metadata extraction from differents parts of scientific documents text such as affiliation extraction, header extraction and bibliography references extraction. In this research project, we make use of references and entities extraction functionality. References and entities extraction functionality is based on several Conditional Random Fields (CRF) models.As shown in Figure 2.6, the CRF citation model includes other CRF models implemented by cascade in order to extract the right entity in the bibliographic reference using the associated CRF algorithm. The evaluation of Grobid was carried out with the reference CORA dataset[35], a publicly available dataset that comprehends more than 1200 training examples for cited references. Grobid showed a reliable level of accuracy 95,7% per citation field and 78.9% per citation instance. Moreover, when it extracts the entities from the bibliography a consolidation step is implemented. The consolidation step is about sending a request to Crossref web service for each extracted citation[8] in order to correct the entities extracted from the reference. If Grobid successfully identifies the title and the year, for example, Crossref will possibly retrieve the correct full publisher's metadata. These metadata are then used for correcting the extracted fields and for enriching the results. Interestingly, the instance accuracy for citations goes up to almost 5%, from 78.9% up to 83.2%. Is possible to use Grobid as a jar library, downloadable from its GitHub repository[2].

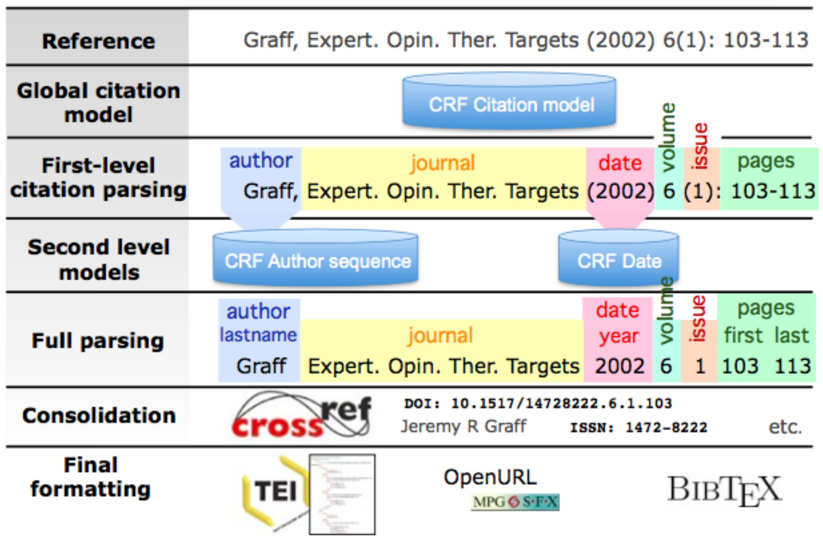


FIGURE 2.6: Work flow of Grobid[20]

2.3.3 Crossref

Crossref API from Crossref [8] works differently than others system mentioned above. As explained before Crossref API was used by Grobid for consolidating its output on the entities extracted from the bibliography. Crossref is an official not-for-profit membership association for publishers, established in 2000. It provides reference linking services for over 62 million scholarly content items such as journal articles, research reports, data sets, and official publications. This is achieved by connecting a unique Digital Object Identifier (DOI) to the metadata about the object, such as a URL, indicating where the object can be found. Crossref works like Scopus database, it does not store full-text scientific content but interlinks millions of items from a variety of content types. Crossref offer severals services, one of these is Crossref API[3]. Crossref API allows querying the Crossref database by giving it in input strings that contain bibliography references. The reference string does not to be necessarily a

[2]https://github.com/kermitt2/grobid
[3]https://github.com/CrossRef/rest-api-doc

well-written references. Moreover, the system allows for input consecutively more references at the same time. At first, the input string is parsed using machine learning techniques, then, the system tries to match the reference string with the metadata that are stored in the database. Another peculiarity of Crossref API, that comes really helpful in this research project, was the score of sureness that Crossref API retrieve beside the document's metadata. For each request, Crossref score indicates how much is sure about the entities retrieved. For example, If the score is higher than 105 means that the metadata retrieved are the corrected ones, otherwise, if the score is lower the metadata retrieved might be the wrong ones.

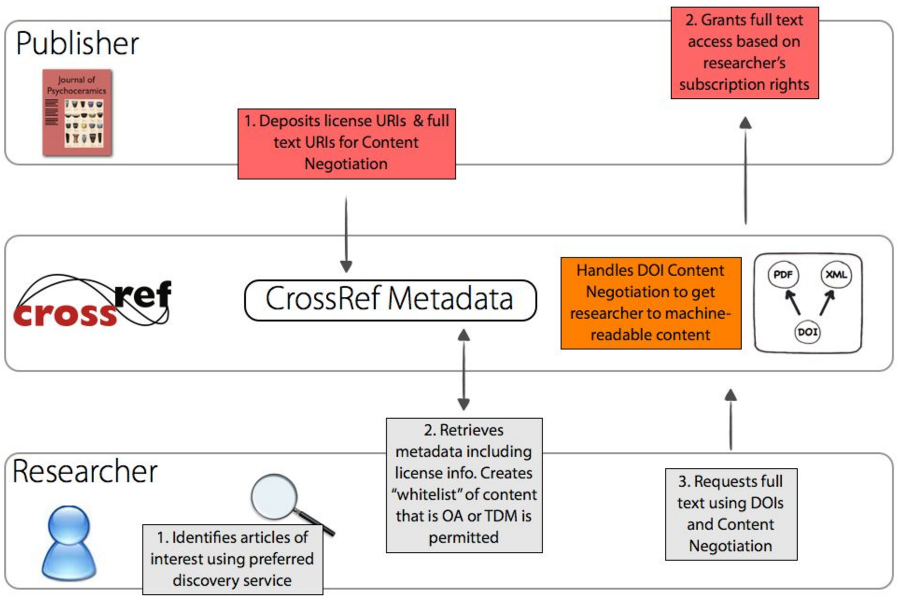

Publisher
Journal of Psychoceramics
1. Deposits license URIs & full text URIs for Content Negotiation
2. Grants full text access based on researcher's subscription rights
crossref
CrossRef Metadata
Handles DOI Content Negotiation to get researcher to machine-readable content
PDF
XML
DOI
Researcher
1. Identifies articles of interest using preferred discovery service
2. Retrieves metadata including license info. Creates "whitelist" of content that is OA or TDM is permitted
3. Requests full text using DOIs and Content Negotiation

FIGURE 2.7: Work flow of Crossref

2.4 Summary

In summary, many efforts were made on entities extraction from bibliography references. Those works are done primarily for helping the scientific community to retrieve information faster and contribute to building better digital libraries. Some works use heuristic approaches, others machine learning techniques, and some of them a combination of both. All of them, at first parse the document to understand the structure of it, later they proceed by extracting the metadata. Especially Cermine and Grobid use machine learning approach that at first parse and categorised the different portions of the scientific document and sub sequentially perform metadata extraction according to the related part of the report. Grobid also uses a consolidation step that involves the use of Crossref API to correct and validate the entities extracted. Those three tools reached almost the perfection when it comes to extracting metadata from scientific structure documents. This research project will test them on entities extraction from unstructured text, and later, their output will be combined to improve their performances and get the right entities in order to retrieve the correct document mentioned in the unstructured text.

Chapter 3

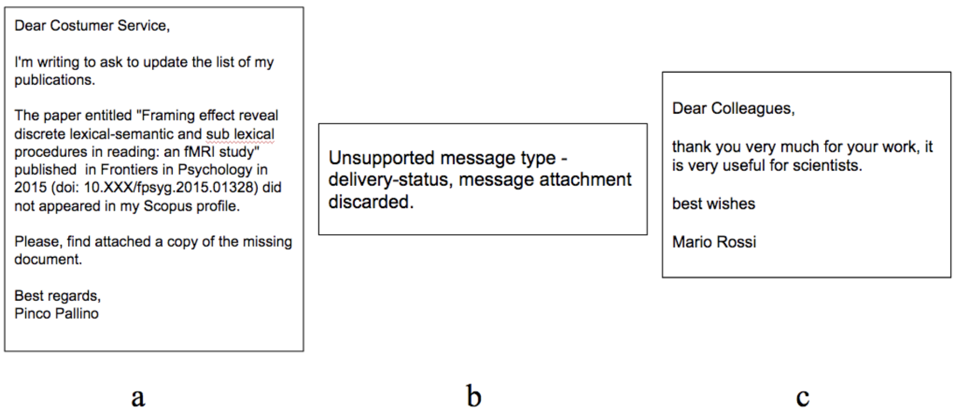
Data

3.1 Customer emails

In order to use machine learning approach to entities extraction from the unstructured text, we have to understand which source of data we are facing and which types of data we can use in order to reach the research project goal. At first, a study on available golden-sets was carried out. There are many golden sets available for free downloading that are used for training systems that perform entities extraction from scientific papers. Cora dataset [35], for example, includes 1878 objects of bibliographical information about scientific articles. Another dataset broadly used for training such systems is UMass dataset [3]. The UMass citation field extraction dataset provides labels and segments for extracted citations from articles found on arxiv.org. On the other hand, unstructured text or free text is considered text that does not follow any predefined structure, hence Cora and UMass datasets are useless for the purpose of this experiment because they store structured bibliography references. Therefore, emails data fit perfectly for our case scenario. Emails are unstructured text, there are no convention that define how the text should be written.
In order to build the reference set, we are going to train and evaluate our system using a sample from the 74.000 emails that Scopus customer service received in 2016. Not all 74.000 emails are usable for this project. The data presented noise, for example, some emails are structured such as:" Unsupported message type - delivery-status, message attachment discarded", others contain just the link of the reference. Therefore, a selection of a good sample is needed, a set of approximately 1000 emails will be selected and annotated using the Brat Annotation tool, an Open source software design for text annotation, that allows for annotation of entities in text segments such as sentences and words [30]. The evaluation set for this experiment is produced by annotating each email according to the snippet of text where the reference is contained. Inside each snippet annotated as a reference, we annotated also the severals elements that compose the reference, the entities such as authors, title, journal and other entities when they were mentioned inside the reference.

3.1.1 Sample of emails selected

Finding the correct way to choose the sample of 1000 emails for annotation was not an easy task. We had to be careful to comprehend all possible cases while choosing our sample because the performance of our system strongly depends on the sample that will be selected. For example, if the sample selected contains just well-written reference the system will be able to recognize and perform entities extraction just on those well-written snippets in the text. On the other hand, if the sample contains too much noise the system will be not able to reach reasonable performances. The optimum case will be selecting those emails from the reference that is written just mentioning few entities to the well written reference. Therefore, we choose to perform our selection by applying a regular expression that was matching if in the email was containing a year. According to the observation that was performed on the emails data we notice that if in the text was contained a year it was probable that the email was including a reference. Therefore, if in the text was contained the string with a year the email was included in our sample, otherwise was discarded. From 74.000 email, slightly more than 20.000 was including a year-string. From those 20.000 randomly 1000 sample was selected for manual annotation.


Dear Costumer Service,

I'm writing to ask to update the list of my publications.

The paper entitled "Framing effect reveal discrete lexical-semantic and sub lexical procedures in reading: an fMRI study" published in Frontiers in Psychology in 2015 (doi: 10.XXX/fpsyg.2015.01328) did not appeared in my Scopus profile.

Please, find attached a copy of the missing document.

Best regards,
Pinco Pallino

Unsupported message type - delivery-status, message attachment discarded.

Dear Colleagues,

thank you very much for your work, it is very useful for scientists.

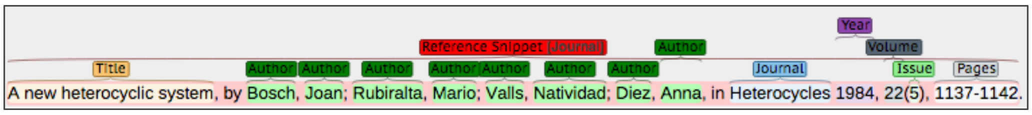
best wishes

Mario Rossi

FIGURE 3.1: Example of emails contained in the 74.000 set.
A: email selected for the reference set, B and C: examples of discarded emails

3.2 Introduction to Brat annotation tool

Manually-curated golden standard annotations are a prerequisite for the evaluation and training of the state of the art tools for most Natural Language processing tasks[30]. Brat annotation tool is a web-based open source tool for text annotation supported by natural language processing technology[30]. The annotation is one of the most time-consuming components of many NLP research efforts; this task had to be precise and curated in order to use the data later in input for training a machine learning algorithm. As an example, from the sample of 1000 email files, it took two weeks to annotate 400 of them. Brat is fully configurable, runnable as a local server, can be set up to help most of the text annotation job such as name entity recognition, dependency syntax and verb frames.



A new heterocyclic system, by Bosch, Joan; Rubiralta, Mario; Valls, Natividad; Diez, Anna, in Heterocycles 1984, 22(5), 1137-1142.

FIGURE 3.2: Entities annotation with Brat

Brat allows marking the span of text by simply selecting it with the mouse by ¨dragging¨ or by double-clicking with the mouse. We used Brat for annotating the references in the text of each email and the subsequent entities that were mentioned.

3.2.1 Entities and references annotated

Each email was analysed manually, and each reference was annotated with its subsequent entities. In total, we annotated 400 emails from the subset of 1000 previously chosen. In this 400 emails, 171 was containing at least one reference. In these 171 files, 452 references were annotated. In some emails, there was mentioned just one reference, and in others, more than one, in our annotated set 39% of the emails contain more than one reference. Moreover, in some references were include all the entities, in others year was missing (258 out of 452 references), not all the authors were mentioned, or title was not completed. In the table below is shown the overall types of entities discovered and marked in our reference set. Overall we annotated 324 Titles, 1116 Authors entities, 194 Journal Titles, 174 Volumes number, 258 Years entities, 81 Doi, 178 Pages-Range entities, 105 Issue number, 25 Publishers, 24 Editors and 30 Issn numbers.

TABLE 3.1: Entities annotated

Entity	Explanation
Title	The title of the scientific document
Authors	The authors that carried out the experiment
Journal	The name of the journal where the article is published
Volume	The number of the volume in the journal
Year	The year when the document is been published
Doi	A digital object identifier, a unique alphanumeric string to identify content and provide persistent link to its location on the internet
Pages	The range of pages of the journal where is been published
Issue	The issue number refers to how many times that periodical has been published
Publisher	The name of the publisher
Editor	The name of the editor
ISSN	The International Standard Serial number that identifies the title of serial publications

Chapter 4

Methods

4.1 Approach motivation

Reference recognition and entities extraction is a studied problem when extracting information from structured text such as bibliography references. In our case scenario, we are facing unstructured text, and this makes our problem similar but slightly more challenging. When we deal with structure text is possible to find patterns due to the several conventions that scientific community use to write a scientific document. On the other hand, the layout and the personalisation that humans utilise when they write a paper, make the content and the style of scientific articles different one of each other, hence, also structure text and documents come in various forms. That's why machine learning approaches are involved in information extraction on structured documents.
This same case we have when it comes to deal with unstructured text. We know that in some part of the text is contained the information that we have to extract but we do not know where exactly is it along the text and which style is written. In other words, the approach that supports the studied experiments can suggest us the methods to follow. Our first goal is to localise those portions of text that contain the piece of information we are looking for: the reference.
In details, the approach followed by most of the experiments previously studied is localise the interested portion of the document and then perform subsequent information extraction. For example, Cermine and Grobid follow exactly this approach when it comes to performing entities extraction from bibliography references. A scientific document has several sections such as abstract, introduction, related works and at the end the bibliography. At first, those systems perform section recognition, the system parses the document, and according to the section, it uses an individual model to extract the right information. For example, when the system recognises the bibliography in the document, it will use a predefined model to get the correct metadata from the references in the bibliography. It is not wise to use the same model for extract information in each section because, as example, the bibliography it has certainly a different structure than the abstract.
This scenario is similar in our case; the reference in the free text is likely to appear in just some parts of the text. As an example, the reference will appear most likely in the body of the email. Therefore, we have to localise at first this part of the text where the reference is contained, as Cermine and Grobid do when they have to extract references from a bibliography in a structured document.
But how we can separate the email chunks in order to perform entities extraction in the right piece of text? To answer this question, we have to understand how we can optimally split our email into parts in order to not lose valuable information. Free or unstructured text, in this case, emails, are written in paragraph or sentences. The reference will appear in one sentence or paragraph, Is very rare that entities of the

same reference are mentioned one separate each other in different parts of the text. This assumption is very useful to our experiment because we know that we can divide the text to keep a reference in just one sentence or snippet.
This research project, then, is divided in two main task. The first task is about narrowing down the problem, splitting the text and classify those snippets as containing the reference or not. The second task will be subsequent entities extraction on those snippets classified as containing the reference.
The implementation of the machine learning classifiers and ensemble algorithms were developed using two different software utilised by NLP community. For what concerns the classification task we made use of Weka [13] a Data Mining Software from the Waikato University that collects machine learning algorithms and tools for data pre-processing, classification, regression, clustering and many others. Weka was adopted for tried out different types of classification in combination with several sets of features. After testing the classification algorithm with Weka, we proceed to implement the models (SVM and Random Forest) with Python and Sklearn toolkit [29] for evaluating them with the whole set of data build with manual annotation. Python in combination with other libraries was used to implement the ensemble models and the evaluation.

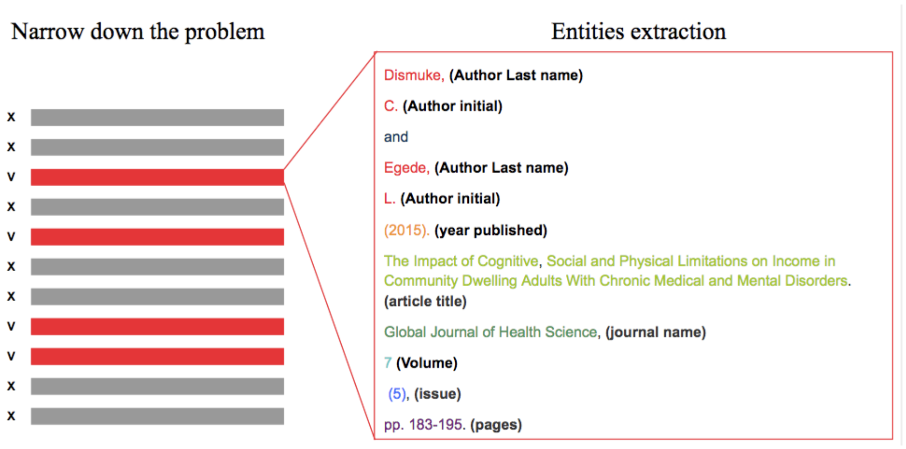

Narrow down the problem
Entities extraction
x
x
v
x
v
x
x
v
v
x
x
Dismuke, (Author Last name)
C. (Author initial)
and
Egede, (Author Last name)
L. (Author initial)
(2015). (year published)
The Impact of Cognitive, Social and Physical Limitations on Income in Community Dwelling Adults With Chronic Medical and Mental Disorders. (article title)
Global Journal of Health Science, (journal name)
7 (Volume)
(5), (issue)
pp. 183-195. (pages)

FIGURE 4.1: Approach for entities extraction

4.2 Reference Sentence Classification

4.2.1 Sentence tokenization

Tokenization, in lexical analysis and text mining, is the process of breaking the text into snippets, a snippet can be a word, a phrase, a single symbol or other meaningful element called tokens. The tokens, then, will be used toward subsequent input for further processing such as parsing or text mining [5]. In this research experiment, the tokens are used in input to "learn" the classifier that will predict if the snippet or token it contains a reference. That’s why the tokenization task is essential. We have to split the text in order to get the optimum snippet that can be used in input to teaching to the classifier what is a snippet that contains the reference and what is not. Therefore, the first problem that we have to face is deciding how to split the text in order to get the proper snippet.
If we split the text too early we con “broke” the information. On the other hand, if we split into wide portion we will have too much noise and our classifier will perform with poor scores.

In other words, if for example, we decide to split the email with a single line, we might be able to recognize the header or the footer of the email but we might break the reference string. Sub sequentially when we are going to extract the entities in the tokenized reference we might be not able to extract all the entities and then will be difficult to retrieve the correct scientific document.
If instead, we are going to tokenize the text into wider snippet we might take in input more the one reference, which can be tricky for two reasons: some parts of the snippet can contain useless information, therefore we are adding noise to our data set. The second problem is the case of having two references inside the same snippet or token. This it will lead to others problem in the next task when we are going to extract the entities, it will be impossible to know which entities belong to which reference.
Therefore, the tokenization of emails in sentences is an essential step for the sake of this experiment. What we aim to reach with this tokenization step is the optimum case where a single reference is contained in a single snippet. In this case, we are able to narrow down the problem and perform a correct entities extraction and subsequent accurate scientific document retrieval. Therefore, five different type of tokenizer were implemented. Each type split the sentence with different rules, each tokenizer was tested and evaluated according to the set of emails annotated. The tokenize sentence was compared with the snippet of the reference annotated by keeping track of the reference string contained in one tokenize snippet.

1. Natural language Toolkit(NLTK) sentence tokenizer in combination with PunktSentence Tokenizer were the first tokenizers implemented. NLTK is a platform for building Python programs to work with human language data, it comes with a suite of text processing libraries [5]. We used PunktSentence Tokenizer which is an instance of the NLTK sentence tokenizer, pertained with a dataset of abbreviations in English language. It is able to recognize whether there is an abbreviation such as Dr. or Mr. in the text, hence, it tries to tokenize the sentence according to punctuation avoiding the punctuation contained in those abbreviations.

2. Regex tokenizer, it makes use of regular expression, a language that use patterns to match on text. Thanks to regular expression is possible to define rules that match portions of text [33] . The regular expression applied to the tokenizer was tokenize the emails by matching a dot follow by a space and a subsequent capital letter.

3. Length tokenizer, tokenize the text considering the total length of the email and then divide the text into 3 equal parts. This 3 parts of the text are then considered as sentences (or snippets).

4. Multiple sentence tokenizer, tokenize the sentence whether a double newline char is matched (“\n\n”) along the text.

Tokenizing the text that defines a sentence is a challenging task. When we consider a paragraph usually the punctuation is essential, indeed, a comma or a dot usually are the boundaries that define a sentence or a paragraph. On the other hand, the reference contains several commas and dot that define the several entities. Therefore the presence of the reference in the text could trick the tokenizer if it just looks at the punctuation in the text.

Tokenizer	Explanation
PunktSentence Tokenizer	Tokenize the text looking at the punctuation avoiding abbreviation
NLTK Sentence tokenizer	Use an instance of PunktSentence Tokenizer and English language from NLTK library
Regex tokenizer	Tokenize the text by matching the dot follow by a space and a capital letter
Lenght tokenizer (1/3)	Tokenize the text by deviding the text 3 equal parts
Multiline tokenizer	Tokenize the text by matching double new line char ("\n\n")

TABLE 4.1: Tokenizer implemented

4.2.2 Preprocessing

Machine learning models approach classification problem by applying mathematical analysis and finding patterns along with data. If the data provided are well structured the model will perform as expected. On the other hand, if the data present noise or are not well prepared, for example, by including mistakes, the model will not predict the correct output. Therefore the quality of data is a crucial aspect when it comes to applying machine learning techniques.
Our problem is about reference recognition; this is a typical classification problem. The model has to be able to recognize whether or not the sentence is containing a reference. We are going to tackle this problem by applying supervised learning. Supervised learning is a type of machine learning algorithm that uses a known dataset also know as training set. The model learns from the training set and then tested on unseen data. The training data includes input data and response values called also ground truth when the data information are provided by direct observation. From it, the supervised learning algorithm seeks to build a model that can make predictions of the response values for a new dataset. A test dataset is used to validate the model for getting the performances. Larger the dataset is higher and accurate the model will be. In the context of our research project, we preprocess each tokenized email. According to the annotation, a file was selected if was including a reference, from a single one to many. Each sentence of the email was marked with a 1 if in the sentence was contain a reference or with a 0 otherwise.

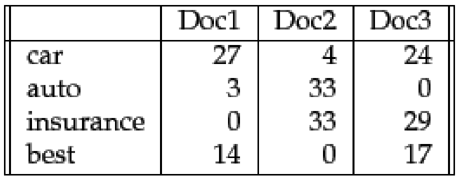
TABLE 4.2: Sentences label

Sentence with references	Title: Nonlinear seismic analysis of unsymmetric-plan structures retrofitted by hysteretic damped braces \nAuthor: Fabio Mazza\nBulletin of Earthquake Engineering\n Volume 14, Issue 4, pages 1311-1331, 2016
Sentence without reference	This paper is already published on the website of Bulletin of Earthquake Engineering (http://link.springer.com/article/10.1007/s10518-016-9873-z)

4.2.3 Description of features

In order to teach a classifier how to predict if a sentence includes a reference or not, we have to convert the original text, in this case, sentences, to a format which can be interpreted by computers and fed into our machine learning model. The sentence classifier is a crucial element of this experiment, hence, we need to propose some features sets and extract as much useful information as possible from each sentence in the email. Text mining provides a collection of techniques that allow us to derive actionable insights from these data. One of the basic features extracted is Bag-of-Words (BoW) model.

In order to get a baseline of our classifier, a Bag-of-Words model was performed on the sentences. Bag of words is an algorithm that counts how many times a word appears in a document. Is applied in text mining mostly to compare documents and gauge their similarities for application like search, document classification and topic modelling. BoW list words with word counts per sentence; the output is a table where the words and sentences become vectors, each row is a word, each column is the sentence and each cell is a word count. Each of the sentences in the corpus is represented by columns of equal length. Those are word count vectors, an output stripped context.



	Doc1	Doc2	Doc3
car	27	4	24
auto	3	33	0
insurance	0	33	29
best	14	0	17

FIGURE 4.2: Example of table for bag of word count on 4 terms in the Reuters collection of 806,791 documents[25]

Not only Bow word level count was performed, char level count was implemented and tested with the classifier. BoW char level works the same as the word but instead of counting by words is counting by char. The sentence is tokenized by chars and each occurrence is vectorized in the table as is we did with the words. Moreover, N-gram takes into account the char that occur after and before the char selected and also those are included in the vectorized table as shown in the table below.

	Doc1	**Doc2**	**Doc3**
ca	15	2	13
ar	10	2	10
au	2	22	17
ut	1	8	10
to	1	3	20

TABLE 4.3: Example of Bow 2-gram for the words *car* and *auto* in the Reuters collection of 806,791[25]

Subsequently to BoW, Term frequency inverse document frequency (TF-IDF) feature model was implemented. TF-IDF is another way to judge a sentence through words and chars contained in the text. The difference between BoW and TF-IDF is that BoW measure frequency and TF-IDF measure relevance, with TF-IDF, words are given weight. First TF-IDF measures the number of times that words appear in a given sentence (like BoW do). Although words such as "and" or "the" appear frequently in the text those are systematically discounted. That is the inverse-document-frequency part. The more a word appears through the sentences the less valuable it is. Each word or character TF-IDF relevance is a normalized data format that also adds up to one. That is intended to leave only the frequent and distinctive words as markers. The word-frequency are calculated with the formula in the figure below, the weight of a word or char *i* contained in a document *j* is given by the number of occurrences *tf* multiplied by the logarithm of the total number of documents *N* divided by the total number of documents that contain the word or char *i*, *tfi,j* is the term frequency of term *i* in document *j*, the number of times that term *i* occurs in document *j*. *wi,j* is the tf-idf weight of word *i* in document *j*.

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

$tf_{i,j}$ = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

FIGURE 4.3: TF-IDF formula

TF-IDF applied in our case scenario it well-fit. If we consider an email, a reference and the combination of both, in the email will use the same language with conjunction, verbs and adverbs. On the other hand when it comes to writing the reference the language change. the case where a sentence contains the reference will appear with terms that are not used in the other part of the emails, hence TF-IDF increase significantly the performance of the classifier.

term	df_t	idf_t
car	18,165	1.65
auto	6723	2.08
insurance	19,241	1.62
best	25,235	1.5

FIGURE 4.4: Examples of idf values on 4 terms in the Reuters collection of 806,791 documents[25]

The third sets of features implemented were syntactic-textual feature. These features know as part-of-speech(POS) tags, aim to recognise whether occur a noun or a verb in the text. The text, or in this case, the sentence, is first split in word token and the nouns, verbs and other syntactic entities are substituted within the text, for example, if a word is a noun it will substitute with "NN" if is a verb with "VV". Then the text in output is analysed by counting the number of occurrence of noun and verbs. In our particular case, we want to find relevance when in a sentence are include more nouns than verbs. Indeed in a reference is more likely to contain noun such as authors name, journals names or cities name than verbs. POS feature was combined with shallow-textual features. Shallow features provide a lot of information that can help our classification task, hence that information can be easily obtained by checking on the text their appearance or counting. In other words, punctuation, for example, is crucial in our case because if we consider a reference is likely to contain a lot of it. Counting comma or dots can boost the performance of our classifier. Several punctuation-based features are implemented. Other features such as year count or capital letters ratio are included. In the table below we can observe all of them.

TABLE 4.4: Textual features

Feature Category	Feature name	Explanation
Shallow-textual features	Paragraph-Lenght	Count the number of words in the setence
	Capital Percentage	The percentage of words that start with capital letter
	Number-count	The count of numbers in the setence
	Dot-count	The number of dot in the sentence
	Comma-count	The number of comma in the setence
	Other punctuation-Count	The number of other punctuation except for comma and dot
	End-with -dot	A boolean value that indicates if the sentence ends with dots or not
	End-with -colon	A boolean value that indicates if the sentence ends with colon or not
	Year-count	A boolean value indicating if a year between 1800 and 2100 exists in the sentence
	Round Brackets count	The number of round brackets in the sentence
	If journal word exists	A boolean value that indicating if the word "Journal" appear in the sentence or not
	If volume word exists	A boolean value that indicating if the word "Volume" appear in the sentence or not
	If article word exists	A boolean value that indicating if the word "Article" appear in the sentence or not
	If issue word exists	A boolean value that indicating if the word "Issue" appear in the sentence or not
	If title word exists	A boolean value that indicating if the word "Title" appear in the sentence or not
	Double quotes count	The number of double quotes that appear in the setence
Syntactic-textual features	NN-ratio	The percentage of words marked as noun
	VV-ratio	The percentage of words marked as verbs

4.2.4 Binary classifier

All these features, mentioned in the previous section, in combination with the labelled sentences are use as input for training a binary classifier. As briefly explained above, the binary classifier is essential for the goal of our experiment, the approach inspired by previous studies, it helps to narrow down the problem by focusing the entities extraction in the portion of the text where exists the reference.In order to train the classifier, the reference set or golden set has to be divided into training data and test data. The training data are the data that the classifier learn from. Instead, the test data are used for evaluating the performances of the model. Therefore the reference set was divided into two part: 80% of it was selected randomly as a training set, the rest 20% as the test set. At first k-fold-cross validation was performed on the entire data sets in order to evaluate the binary classifier. K-fold-cross validation is a technique to evaluate predictive models by splitting the original dataset into a training set to train the model, and a test set to evaluate it. It is often applied to get accurate performances with a limited amount of data. The data were divided into *k* subsets, the train and the test are repeated *k* times. Each time, one of the *k* subsets is used as the test set; the other *k-1* subsets are put together to form a training set. According to the size of our reference set, we evaluate the model using 5-fold cross validation. When we reach respectable performances, we trained the classifier with all the 80% of data and we tested it with the 20% unseen data selected as a test set. Two different classifiers were compared Random Forest and Support Vector Machines(SVMs).

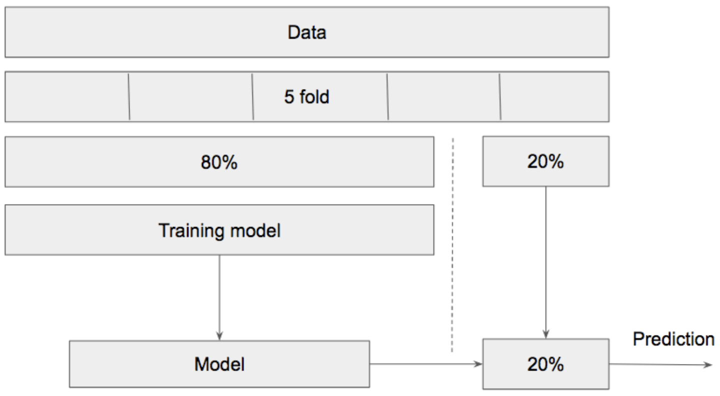


FIGURE 4.5: Binary classifier work flow

4.2.5 Random Forests

Random Forest or Random Decision Forest is considering as an ensemble learning method for classification and regression. Is it studied that the combination of learning models increases the classification accuracy, this method is call Bagging (or bootstrap aggregating). Random Forest has multiple Decision Tree, each of it performs a particular output that will be combined with others Decision Tree the output that receives more votes will be selected as the true one. In other words, the main idea of Bagging is to average noisy and unbiased models to create a model with lower variance. Therefore, Random Forest algorithm works as an extensive collection of correlated decision trees. Random Forest, it creates multiple Decision Trees and then combines their output for the best one. In the matrix below is shown the aim of random forest. Let's suppose that the matrix *S* is a matrix with training sample that we submit to the algorithm to create a classification model, the *fa1 ... fan* is the feature extracted from the text. For example, *fa1* is the feature a for the first sample and again the *fbn* is the feature *B* of the *Nth* sample. The *C*, in the last column, is the training class, in this research project, the label of the sentence is *1* if it contains a reference *0* otherwise.

$$S = \begin{bmatrix} f_{A1} & f_{B1} & f_{C1} & C_1 \\ \vdots & & \vdots & \\ f_{AN} & f_{BN} & f_{CN} & C_N \end{bmatrix}$$

FIGURE 4.6: Matrix decision tree

From this sample set, random forest it creates many others subsample with random values. For example, as shown in the figure below, the sample 1 it takes the elements 12, 15, 35, and again in the sample 2 it selects the elements 2, 20, 6 and other random elements. From each of those subset, random forest it creates a decision tree. Each of these decision tree has its own output, which with each of this output will create a decision ranking.

$$S_1 = \begin{bmatrix} f_{A12} & f_{B12} & f_{C12} & C_{12} \\ f_{A15} & f_{B15} & f_{C15} & C_{15} \\ \vdots & & \vdots & \\ f_{A35} & f_{B35} & f_{C35} & C_{35} \end{bmatrix} S_2 = \begin{bmatrix} f_{A2} & f_{B2} & f_{C2} & C_2 \\ f_{A6} & f_{B6} & f_{C6} & C_6 \\ \vdots & & \vdots & \\ f_{A20} & f_{B20} & f_{C20} & C_{20} \end{bmatrix}$$

$$S_M = \begin{bmatrix} f_{A4} & f_{B4} & f_{C4} & C_4 \\ f_{A9} & f_{B9} & f_{C9} & C_9 \\ \vdots & & \vdots & \\ f_{A12} & f_{B12} & f_{C12} & C_{12} \end{bmatrix}$$

FIGURE 4.7: Matrix Random forests

In other words, each decision tree has an output of the predicted class, the output will be ranked per the majority of the votes of each decision tree. For example, if the Random Forest is composed of four decision tree as the figure below show us, the prediction of each decision tree will be combined and the class that take more votes is the one selected.

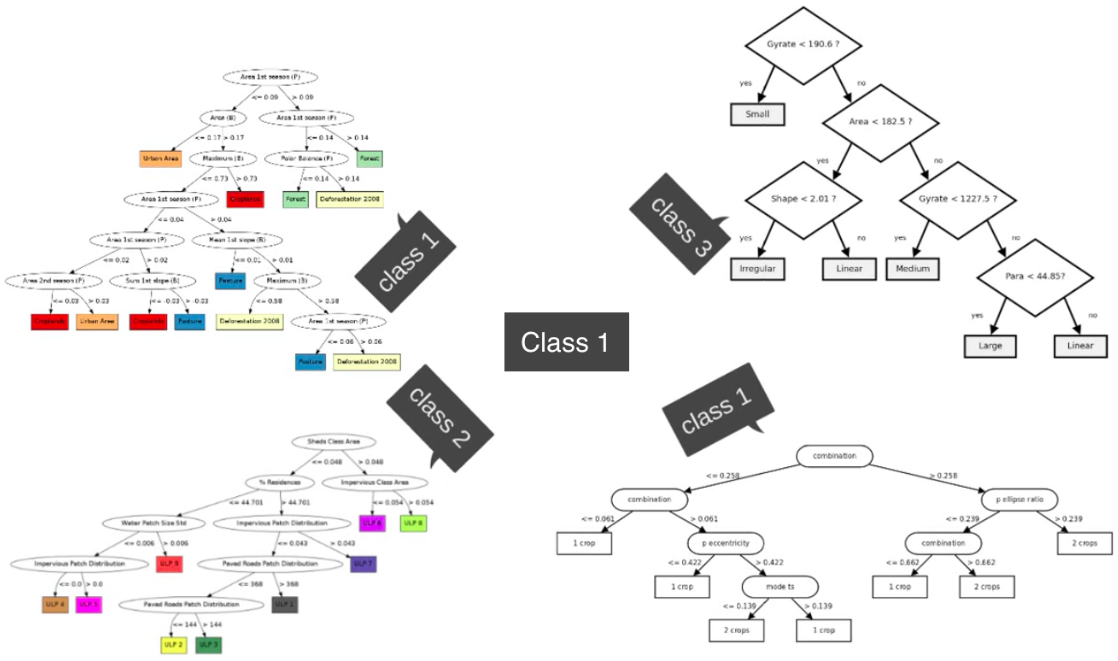


FIGURE 4.8: An example of bagging

Random Forest classification algorithm was implemented because it takes the groups of features and tries to apply them each one separately so the features they do not depend on one to each other. When it comes to combining features TF-IDF, syntactic-textual feature and shallow-text features, we wanted to understand if the classifier was deducing better by considering the features one independent to the others as in the case of Random forest or when the features were considered dependent on each other in the case of Support Vector Machine (SVM).

4.2.6 Support vector Machines

A support vector machine (SVM) is machine learning algorithm that analyzes data for classification and regression analysis. SVM is a supervised learning method that looks at data and sorts it into one of two categories. It is a discriminative classifier formally defined by a separating hyperplane [15]. In this algorithm, we plot each data item as a point in n-dimensional space (where n is a number of features) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes. Suppose some given data points each belong to one of two classes, the goal is to decide which class a new data point will appertain to. In support vector machine, a data point is a view as a p-dimensional vector, and the solution will find the line called hyperplane that separates the two classes. For example, as shown in the figure below we have some separable binary sets, we have *x1* and *x2* feature and we want to classify if an element is a square or a circle. In our real scenario, will be the sentence classify as contain the reference or not.

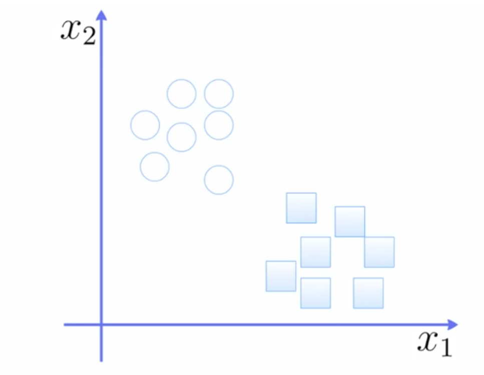


FIGURE 4.9: Example of two classes

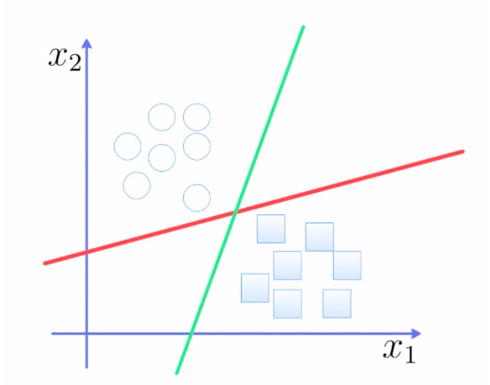


FIGURE 4.10: Example of possible hyperplane

The goal is to design a hyperplane that classifies all training vectors into two classes, find the line that separates the elements in the plot. The best choice to draw the line that leaves the maximum margin from both classes. By margin, we mean the distance between the line and the closest element as mark as *z1* and *z2* shown in the figure below.

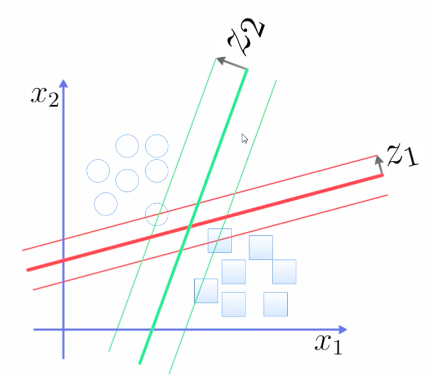


FIGURE 4.11: Margin of hyperplane

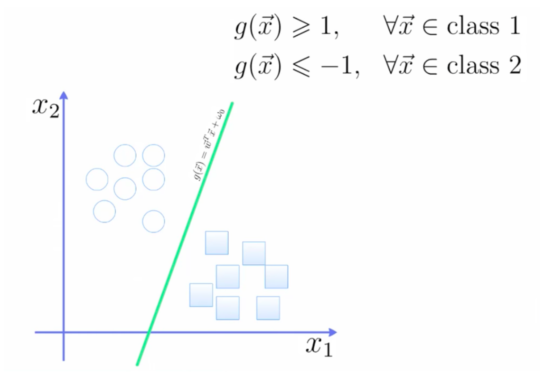

$g(\vec{x}) \geqslant 1, \quad \forall \vec{x} \in$ class 1
$g(\vec{x}) \leqslant -1, \quad \forall \vec{x} \in$ class 2
x_2
x_1

FIGURE 4.12: Classification

In this example, we can clearly see that *z2* is grater then *z1*, hence, the margin is higher in the case of the green hyperplane. The hyperplane is defined by an equation. This equation will define the boundaries of the class and will predict if a new data point is considered a circle (class 1) or a square (class 2). This works the same with our sentences except that our features dimensionality is more higher. If you want have a deeper explanation on how SVMs works, please refer at this document [31].

4.3 Ensemble for entities extraction

In summary, at first the text is tokenized in sentences. From each sentence we extract the features that we use as input to the classifier. The binary classifier will predict if the sentence is containing a reference or not. If the classifier finds a reference in the sentence the next step will be performed entities extraction to retrieve the correct paper mentioned in the unstructured text. By taking inspiration from previous studies we wanted at first evaluate such systems that performs entities extraction from structured text. Those tools that use machine learning approaches are learned with several types of data and they were developed aiming to adapt to several layouts. The sentence predicted as contain the reference, hence, is parsed from Cermine[34], Grobid[20] and Crossref API [8] order to get their output on the sentence contacting the reference. Those systems were performing quite good according to the entities that was extracted. We observed that Cermine was performing well when it comes to recognize entities that contain number such as year, pages, volume number. On the other hand, Grobid was better on Authors recognition, Crossref instead was retrieving the right paper when in the sentence contain the reference with the entities well mentioned. As Random Forest teach us we decide to create our own system to get better performance on entities extraction from unstructured text by combining the output of those three systems. This methodology is called ensemble and is a technique that combine the models to produce improved results. Ensemble methods usually produces more accurate solutions than a single model would. After the normalization on each system output, two model was produced. The first one, by simple bagging or majority vote, the second one is a smarter version of the first one that essentially gives weight of each output according to the results observed during the evaluation of the three systems. The two model was, then, evaluated by comparing them with the others systems showing interesting improvements.

Ensemble Majority

The input of the first model is the output of Grobid-Cermine-Crossref. To combine their output, we developed an algorithm that normalizes the entities extracted from each system and by using the similarity between strings, it combines the entities into one single output. In other words, the sentence classified as a reference is given in input to Grobid, Cermine and Crossref. Each system gives back as output the entities that it recognized in the sentence. The algorithm at first combine the entities extracted in one single array grouped by type of entity. If the same entity is recognized by more the one system, is taken as the true entity otherwise is discarded.

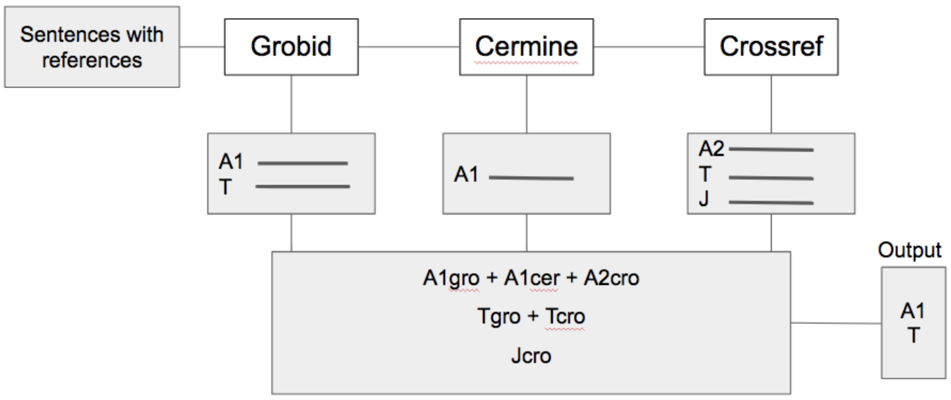


FIGURE 4.13: Ensemble work flow majority

Ensemble Majority Weighted

The second model is smarter. Was observed that between that system Cermine was performing better on extracting entities that contain numbers (Chapter 5) such as pages, year, and volume. On the other hand, Crossref is able to extract a multitude of entities and overall its performance is slightly better than the others. Moreover, it gives also the relevance score. The relevance score is the percentage of the level of sureness on the retrieved entities. Was observed that if the score is higher than 105 the retrieved entities are the correct ones. Inspired by those results, the models by the majority was upgraded. When it comes to extracting numeric entities an higher weight to the output of Cermine is given. On the other hand, when the relevance score of Crossref is higher than 105 the model returns just the entities that Crossref retrieve.

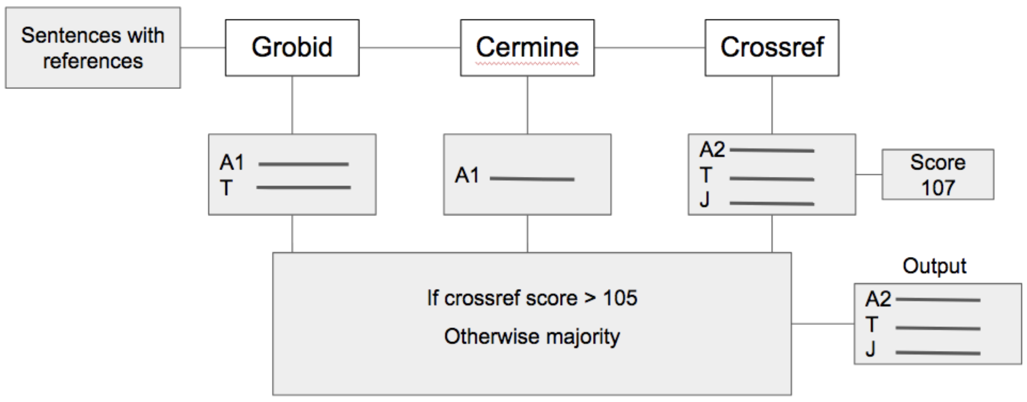


FIGURE 4.14: Ensemble work flow weighted

Chapter 5

Results

In this chapter, we presents the results that we got from all the sub-experiments that were carried out in order to reach the final research goal of entities extraction from unstructured text. We had, at first, to find a way to tokenize in snippets the unstructured text. Five different types of tokenization were performed and the results are shown in the tables below in this Chapter. By tokenize the free text we aim to split the text in a manner that a reference was contained in a single snippet. In this way, we could classify each sentence as containing the sentence without excluding any information. On this purpose, we developed a binary classifier. We had to extract some feature form each snippet in order to teach, to the binary classifier, how a reference in unstructured text looks like. We carry out four different type of features and apply them first individually and then together. We try two types of binary classifier Support Vector Machines(SVMs) and Random Forest. Scores are given for both the models, applied to the four sets of features. When the sentence was predicted as containing a reference was given in input at first to the three systems, Grobid, Cermine and Crossref in order to perform entities extraction, the consequent results are assumed in this chapter. Finally, the output of those tree systems is combined and two models are carried out with the consequent results explained below.

Results are given in two different formats; sentence tokenization results are given through percentage of references recognized per sentence. Subsequently, machine learning techniques are applied, hence, the results are expressed by recall, precision and f1-score.

F1-score is used in statistical analysis to measure the accuracy of the system. It considers both the precision and recall measures of the test to compute the score. If we consider our classification problem, we have to classify if a sentence is a reference or not. For example, let's consider that our dataset is a list of 100 sentences, half of which are containing a reference and the other half that does not contain it, we have to give back this list of all 50 sentences that are including a reference, but being careful to not accidentally include sentences that does not have it.
Having high precision means that when you do say that some sentences do contain a reference, you are usually right about it. This is about how many sentences in the list are actually containing the reference, out of all the ones that are returned in the list.

$$Precision = \frac{TP}{TP + FP}$$

Having high recall means that you can identify most of the sentences that contain the reference. This is about how many sentences that include a reference are added to the list, out of all the ones that exist.

$$Recall = \frac{TP}{TP + FN}$$

These two calculation are not the same if the list contains just one single sentence that has a reference we will have very high precision since the only sentence listed is actually containing a reference. On the other hand, we will have very low recall: there are other 49 sentences that are including a reference. But in the list, there just one retrieved. Ideally, the optimum case would have a list with all the sentences that include a reference while being careful to not accidentally include some which are not. If we reach that we would have both high precision and recall.

When we are measuring how well we are doing, it is often useful to have a single number that describe the performance. We could define a number to be, for instance, the mean of precision and recall. This is exactly what F1-score is.

$$F1 = 2 * \frac{1}{\frac{1}{recall} + \frac{1}{precision}} = 2 * \frac{precision * recall}{precision + recall}$$

The only reason why we use the harmonic mean is because we're taking the average of ratios (percentages), and in that case the harmonic mean is more appropriate than the arithmetic one.

5.1 Performance sentence tokenization of email

	References found per snippet (Tot references: 452)	sentences with reference	sentences without referenc
PunktSentence Tokenizer	186 (41%)	163	2253
NLTK Sentence Tokenizer	145 (32%)	105	2173
Regex Tokenizer	194 (42%)	169	2477
Lenght Tokenizer (1/3)	356 (78%)	150	243
Multiline Tokenizer	424 (93%)	343	1488

TABLE 5.1: Text tokenization

The sentence tokenization was the first task performed for this experiment. As mentioned in the previous chapter four different type of tokenizer were implemented, in the table above show the results that each tokenizer reach. The results show respectively the number of total references from the annotated dataset 452, the total references found by tokenizing the text with respectively NLTK PunktSentence tokenizer, NLTK sentence tokenizer, regex tokenizer, length tokenizer and multi line tokenizer which is the one that was performing better than the other.

5.2 Binary classifier for sentences and features performance

Four types of features are extracted from each tokenized sentence. Each group of features is evaluated separately. Then a combination between TF-IDF, shallow and syntactic textual is also given.

The binary classifier takes in input the sentence extracted from the tokenization performed in the previous task, features are then extracted from the sentence and give in input to the binary classifier that it tries to predict if is the sentence contain a reference or not. Two binary classifiers were implemented SVM and Random Forest. The scores show that SVM along with TF-IDF features performs better than the other.
In this task, the goal was about reaching a better recall than precision. With high recall, we are sure that all the possible sentences containing references are retrieved. It better to perform entities extraction on sentences missed classified as containing reference instead to lose information by missed classify sentences containing a reference.

TABLE 5.2: Binary classifiers performance

	SVM			**Random Forest**		
Features	Recall	Precision	**F-Score**	Recall	Precision	**F-Score**
Bag of words	0.9430	0.7464	**0.8333**	0.6981	0.9585	**0.8078**
Shallow-Syntactict textual features	0.8787	0.9235	**0.9006**	0.8275	0.9523	**0.8856**
TF-IDF	0.91044	0.9472	**0.9284**	0.6935	0.9347	**0.7962**
TF-IDF+textual	1.0	0.9516	**0.9752**	0.8709	0.9890	**0.9262**

5.3 Ensemble on entities extraction

Once that the sentence was predicted as contain a reference, is given in input to the ensemble method that combines the output of Cermine, Grobid and Crossref, in a single one. Therefore, the performances of those systems are shown in comparison with the models implemented that use ensemble method by majority votes and by weighted votes. In this task, we aim to reach good accuracy in order to be sure to retrieve the corrected document, mentioned in the email. For each system, two performances table are given, respectively the first one shown the score of recall, precision and F1-score, by counting the overall number of true positive, false positive and false negative, also known as the macro score. The second table is calculated by averaging the recall, precision and F1-score of each sentence, micro score.

TABLE 5.3: Total scores

Macro score

	Precision	Recall	**F-Score**
EnsembleWe	0.8776	0.7036	**0.7810**
EnsembleMa	0.9519	0.6193	**0.7504**
Crossref	0.7122	0.7122	**0.7122**
Cermine	0.6868	0.4666	**0.5844**
Grobid	0.6396	0.5573	**0.5956**

TABLE 5.4: Total scores per average on sentences

Micro score

	Precision	Recall	**F-Score**
EnsembleWe	0.7297	0.6051	**0.6387**
EnsembleMa	0.7744	0.5381	**0.6156**
Crossref	0.5624	0.5624	**0.5624**
Cermine	0.6251	0.4647	**0.5018**
Grobid	0.6399	0.4488	**0.5058**

In the following sections the performance of the systems tested are shown. Per each entity extracted the precision, recall and F-score are given. For each score table, is also explained the amount of data used for each entity to carry out the performance evaluation.

5.3.1 Title extraction performance

TABLE 5.5

Title

	Precision	Recall	**F-Score**
EnsembleWe	0.9130	0.9545	**0.9333**
EnsembleMa	0.9705	0.8988	**0.9333**
Crossref	0.7083	0.7083	**0.7083**
Grobid	0.7058	0.7755	**0.7390**
Cermine	0.6402	0.9741	**0.7726**

Title extraction evaluation was performed with total 324 titles appearing in 338 sentences containing references, both of ensemble methods majority and weighted perform remarkably good reaching 0.93 of correct title extraction from unstructured text.

5.3.2 Authors extraction performance

TABLE 5.6

Authors

	Precision	Recall	**F-Score**
EnsembleWe	0.8791	0.7132	**0.7875**
EnsembleMa	0.9294	0.7078	**0.8036**
Crossref	0.7888	0.7888	**0.7888**
Grobid	0.8463	0.7401	**0.7896**
Cermine	0.8647	0.3239	**0.4713**

Authors extraction evaluation was performed with 1116 authors in our reference set. Surprisingly the majority method scored slightly better than the others, with 0.80 of F-score. Crossref, Grobid and weighted methodology are also performing great with 0.78 of F-score.

5.3.3 Journal extraction performance

TABLE 5.7

Journal

	Precision	Recall	**F-Score**
EnsembleWe	0.9290	0.8397	**0.8821**
EnsembleMa	0.9561	0.6770	**0.7927**
Crossref	0.7070	0.7070	**0.7070**
Grobid	0.5103	0.5963	**0.5500**
Cermine	0.5820	0.7692	**0.6626**

Journals title extraction evaluation was performed with 194 Journals titles. The weighted method performed better than the others systems reaching 0.88 of F-score.

5.3.4 Volume extraction performance

TABLE 5.8

Volume

	Precision	Recall	**F-Score**
EnsembleWe	0.8771	0.7246	**0.7936**
EnsembleMa	0.9560	0.5878	**0.7280**
Crossref	0.7441	0.7441	**0.7441**
Grobid	1.0	0.0394	**0.0759**
Cermine	0.605	0.8768	**0.7159**

Volume extraction evaluation was performed with 174 volume numbers annotated in our reference set. The weighted method performed better than the others systems reaching 0.79 of F-score. Also, the others systems perform relatively well. Cermine reaches 0.74, the majority votes method 0.72 and Cermine 0.71 of F-score.

5.3.5 Year extraction performance

TABLE 5.9

Year

	Precision	Recall	**F-Score**
EnsembleWe	0.9642	0.9000	**0.9310**
EnsembleMa	0.9890	0.8372	**0.9068**
Crossref	0.6774	0.6774	**0.6774**
Grobid	0.7663	0.7557	**0.7610**
Cermine	0.7354	0.7846	**0.7592**

The year extraction evaluation was carried out with 258 years annotated in our reference set. On this extraction, we reached satisfying results, the weighted methods and the majority one got respectively 0.93 and 0.90 of F-score.

5.3.6 DOI extraction performance

TABLE 5.10

DOI

	Precision	Recall	**F-Score**
EnsembleWe	1.0	0.8961	**0.9452**
EnsembleMa	1.0	0.8961	**0.9452**
Crossref	0.7012	0.7012	**0.7012**
Grobid	1.0	0.0649	**0.1219**
Cermine	0.8837	1.0	**0.9382**

Doi extraction evaluation was performed with 81 annotated doi entities. Doi is pretty easy to recognise because his structure is unique and standardized. Therefore we reached remarkable results. Both majority and weighted ensemble methods got 0.94 of F-score. Another remarkable score is the 0.93 F-score of Cermine.

5.3.7 Pages extraction performance

TABLE 5.11

Pages

	Precision	Recall	**F-Score**
EnsembleWe	0.8547	0.7142	**0.7782**
EnsembleMa	0.9870	0.4871	**0.6523**
Crossref	0.6569	0.6569	**0.6569**
Grobid	0.5348	0.2690	**0.3579**
Cermine	0.7192	0.8541	**0.7809**

Page extraction evaluation is performed with 178 page entities marked in the reference set. Surprisingly the system that reached more accuracy is Cermine with 0.78 of F-score.

5.3.8 Issue extraction performance

TABLE 5.12

Issue

	Precision	Recall	**F-Score**
EnsembleWe	0.7872	0.4404	**0.5648**
EnsembleMa	not retr.	not retr.	**not retr.**
Crossref	0.6800	0.6800	**0.6800**
Grobid	not retr.	not retr.	**not retr.**
Cermine	not retr.	not retr.	**not retr.**

Issue extraction evaluation was performed with total 105 annotated entities. Unfortunately not all the systems were able to recognize the Issue in unstructured text, this influenced the performance of the others systems. Grobid and Cermine didn't retrieve the entity, hence, the majority method couldn't retrieve it as well.

5.3.9 Publisher extraction performance

TABLE 5.13

Publisher

	Precision	Recall	**F-Score**
EnsembleWe	0.6153	0.4705	**0.5333**
EnsembleMa	1.0	0.0952	**0.1739**
Crossref	0.2272	0.2272	**0.2272**
Grobid	0.4827	0.5600	**0.5185**
Cermine	not retr.	not retr.	**not retr.**

Publisher extraction evaluation was performed with total 25 annotated entities. Due to the number of entities annotated the performances are not higher. The system that performs better is the weighted one with 0.53 of F-score.

5.3.10 Editor extraction performance

TABLE 5.14

Editor

	Precision	Recall	**F-Score**
EnsembleWe	1.0	0.2222	**0.3636**
EnsembleMa	0.0	0.0	**0.0**
Crossref	0.0	0.0	**0.0**
Grobid	0.075	0.3333	**0.1224**
Cermine	not retr.	not retr.	**not retr.**

Editor extraction evaluation was performed with total 24 annotated entities. Same as for the publisher, due to the number of entities annotated the performances are not higher. The system that performs better is the weighted one with 0.36 of F-score.

5.3.11 Issn extraction performance

TABLE 5.15

Issn

	Precision	Recall	**F-Score**
EnsembleWe	0.7333	0.5789	**0.6470**
EnsembleMa	not retr.	not retr.	**not retr.**
Crossref	0.4642	0.4642	**0.4642**
Grobid	not retr.	not retr.	**not retr.**
Cermine	not retr.	not retr.	**not retr.**

Issn extraction evaluation is given according to the 30 annotated ISSN entities in our reference set. Not all the systems were able to recognize the ISSN entity because usually doesn't occur in structured references.

Chapter 6

Discussion

In this chapter, we will discuss the results obtained on this research project. Firstly, we will have some discussion about the tokenization on the text. Secondly, we will discuss the obtained scores of the binary classifiers with the four set of features extracted, on both models SVM and Random Forest. Finally, we discussed on the ensemble method and the performances achieved while performing entities extraction.

6.1 Discussion on sentence tokenization results

The first challenge addressed was the email text tokenization. As mentioned above along the report we implemented four different kinds of tokenizer. At first, we made use both tokenizer implemented by NLTK Sentence tokenizer and PunktSentence tokenizer. Unluckily we obtained bad performances, indeed, respectively 32% and 41% of the total references were tokenized correctly (an entire reference per sentence). Although PunktSentence tokenizer and NLTK Sentence tokenizer are broadly used and praised from the NLP and text mining community, in our case had poor results because we are dealing with text that contains references, and a reference usually include a lot of punctuation. Even though both tokenizers are trained for escape punctuation for abbreviation such as Dr. or Mr., they are not trained for escape punctuation contained in the reference used to divided the entities such as the comma or dots that are used to separate the Authors names. We observed that by applying strong rules on the tokenization, poor results were obtained. When we applied the Regex tokenizer the score achieved was about 42%.
According to the results obtained with those tokenizers, we decide to apply wicker rules. Length tokenization was implemented. reaching 78% of correct tokenization. Finally, we tried to split the text whenever a double newline char was encountered ("\n\n"). Out of overall 452 references, 424 was correctly tokenized (93%). This result indicates that most of the customers start a new paragraph when they are about to write the reference in the email. When it comes to including the reference in the text, usually the customer press double times the enter button recognizable by the computers with the sequence of chars "$\backslash n \backslash n$".Although we reach satisfying performance by applying this kind of tokenization, it can be tricky when it comes to recognize and extract entities from other types of unstructured text such as social networks posts. This style of writing it will be likely to appear most on emails text than social networks post, for example. A solution will be applying the PunktSentence tokenization. The PunktSentence tokenizer is an unsupervised trainable model, this means that it can be trained for skipping another kind of punctuation. By training the PunktSentence tokenizer to skip reference punctuation we will be able to split the sentence in a better method, reaching better scores than Multiline tokenizer.

6.2 Discussion on Binary classifier results

From the performance, shown in Chapter 5 we can evaluate the performances of two different classifiers that was implemented along with the set of features of Bag-of-Words, TF-IDF, shallow-textual and syntactic-text features. From the results, we can observe that the combination of TF-IDF and shallow-textual and syntactic-text features was performing better than the others.
Bag-of-Words were the first set of feature that was implemented. It was implemented to get a base line score on our preprocessed data. According to the observation of the data, we produce the shallow and syntactic textual features reaching better performances that our baseline, 83% against 90% of F1-score. Textual features are still too general for proper understand unstructured text, although 90% of f1-score still consider an acceptable performance we wanted to obtain even better performances since the classification process is essential for achieve good results on the entities extraction task, our research goal, .
Therefore, as briefly mentioned in Chapter 4, we thought that email text is likely to contain the same terms when we are not referring to a reference. On the other hand, when we are mentioning the reference the terms changes. Therefore, IT-IDF improved the performances of the classifier up to 92% F1-score.
We proceed a combination of those last sets of features. The main object of this classification task was to reach as much recall as possible, in order to be sure to perform entities extraction on all the possible sentences. Although a sentence can be miss-classify as contain references even if it does not, the ensemble model will return no entities if in the sentence there is no one. On the other hand, lose information by miss-classify a sentence with reference, as a sentence without reference, is certainly worst.
Two different classifiers were evaluated and compared: SVM and Random Forest.
SVM classifier shows better performances than Random forest. This is due to the high dimension that we use with the features. Bag-of -Words and TF-IDF are high-dimensional sparse data because SVM classifiers are better suited to high dimensional data than the Random Forest. This is the reason why we obtained better performances applying SVMs than Random Forest.

6.3 Discussion on ensemble results

We will proceed by discussing the scores on entities extraction. The We started by evaluating the performance of Grobid, Cermine and Crossref. The evaluation was not an easy job, all their output when the entity is extracted, come out in several formats. For example, Grobid retrieve the author entity by separating the names in different arrays, Crossref retrieves them in well structure dictionary [28], Cermine, instead returns a simple string. Therefore a normalization step was needed in order to evaluate the systems in the correct way. Another difficult task was the evaluation of Crossref. As you may notice the scores of Crossref are equal on recall and precision. This is due to the same amount of false positive and false negative that it collected during the evaluation process. Crossref, indeed, when it processes the string with the references it does not try to extract the entities directly from the string but it parse it and sub sequentially retrieve all the entities that are stored in its database. If it gives back the correct paper than it will have all true positive entities, and when it returns the wrong one it will have all false positive for all the wrong entities that it retrieves with the same amount of false negative for the entities that it could not

guesses.
Two models were carried out with the ensemble method. The first model use bagging or majority votes, where the votes are treated with the same weights, hence, by applying majority we reach a good result in term of precision, less in term of recall, because the model was taking as true entity when all the systems, or the majority, was returning the same one. The second model, act with a slightly smarter algorithm the votes are weighed according to the observation made during the testing process. In this manner, we could raise the recall and get better performance.

Chapter 7

Conclusion and Future works

In this final report, we illustrate the process to perform entities extraction from unstructured text, including text tokenization, sentence classification and entities extraction. With this research project, we wanted to show an end-to-end process starting with raw data and finishing with the correct entities extraction, in order to retrieve the right document mentioned in the unstructured text.
In the coming "Conclusion" section, we explain how our research questions are answered. And at the end, we describe some future works that can be developed and can probably improve the performance of the system.

7.1 Conclusion

For our main research question, *"Given unstructured text containing scientific articles references, how can machine learning approaches perform on the task to recognise and extract reference entities?"*. The answer has been given through the approach that has been followed in the research. The problem has been narrowed down to perform entities extraction to the piece of text that is containing the reference.
There were two crucial steps for being able to recognise the reference in the free text, and to be able to answer the first subquestion which was: *"How can we identify a snippet in the unstructured text where the reference contains itself?"*. Firstly we applied text tokenisation, being careful not to break the reference while dividing the text into snippets, and later performed reference recognition on those portions of the text. By devideing the text into snipetts containing entire refernces, It is possible to brake the text in order to train a machine learning classifier. We showed that by combining textual and TF-IDF features, it was possible to receive great performances on the sentence classification. Both precision and recall were necessary for correctly classifing a sentence as containing a reference. But with a high recall, we are sure to consider all the possible sentences that can include a reference. For example, if a sentence is without a reference but is classified as containing a reference, in the next step when the sentence is parsed for entities extraction, no reference's component will return. When the snippet or sentence in the text has been classified as containing a reference, we had to extract the several entities to be able to retrieve the right scientific document.
We answer, hence, to the second sub-question: *"How can we identify and extract reference entities contained in a snippet of unstructured text?"*. Firstly, we tested the three pre-trained systems, Grobid [20], Cermine [34] and Crossref [8], that performs entity extraction on structured text such as bibliography and string references. All of these systems are implemented with machine learning techniques that can adapt to several types of text structures. We observed that they were able to extract reference entities sufficiently and we opted for a model that was combining their output, also called ensemble method, to obtain a better performance. Two models were

produced. The first one connects the different output by majority votes. With this method we reached good precision, beating the others systems on almost all entities extraction from free text.
The second model is slightly smarter, according to the observation of the entities extraction performances, obtained by the three systems Cermine, Grobid and Crossref and considering the "sourness" score that Crossref retrieved, votes were weighted. In this way, the recall was increased, and better performances in term of F-score was achieved.
This research project is a starting point, and it will lead to automating Scopus Customer service tasks. In order to fully automate tasks such as identification of "out of policy" documents, or documents that are still in the production phase (and therefore not referenced in Scopus database), the machine, at first, has to recognise the scientific document mentioned in the free text or for the case of Scopus customer service, in emails. The good performance, in terms of accuracy, on entities recognition shows the value of this methodology, followed in this research project.

7.2 Future works

By applying machine learning on unstructured text containing scientific references, we can extract the entities with great accuracy.
On the other hand, several aspects of this system can be improved. Firstly, more data are needed for gain a better evaluation on those entities that appear less frequently in our reference set. Another major step that can considerably improve the performance of the system is the tokenization phase.
Even though we could reach good performances on our data by tokenizing the text whenever was occurring a multiline character (“\n\n”) this scenario typically happen when we consider emails text. When we are writing an email, for example, we are used to splitting the sentences or paragraph by including an empty line. The same occurred in this case; when the Author was about to add the reference in the email was pressing double time the "enter" button and then was including the reference. On the other hand, when it comes to deal with other unstructured text such as web forms or social media post we will not have the same case because usually the character and space are limited. Therefore, NLTK PunktSentence tokenizer [5] can be trained to escape such as punctuation that occurs in the reference, hence, tokenizes the text in a better manner.

Bibliography

[1] Ethem Alpaydin. *Introduction to machine learning*. MIT press, 2014.

[2] Modern Language Association of America. *MLA handbook for writers of research papers*. Modern Language Association of America, 2010.

[3] Sam Anzaroot and Andrew McCallum. "A new dataset for fine-grained citation field extraction". In: (2013).

[4] American Psychological Association et al. *Publication manual*. American Psychological Association, 1983.

[5] Steven Bird. "NLTK: the natural language toolkit". In: *Proceedings of the COLING/ACL on Interactive presentation sessions*. Association for Computational Linguistics. 2006, pp. 69–72.

[6] Johan Bollen, Huina Mao, and Xiaojun Zeng. "Twitter mood predicts the stock market". In: *Journal of computational science* 2.1 (2011), pp. 1–8.

[7] Alexandru Constantin, Steve Pettifer, and Andrei Voronkov. "PDFX: fully-automated PDF-to-XML conversion of scientific literature". In: *Proceedings of the 2013 ACM symposium on Document engineering*. ACM. 2013, pp. 177–180.

[8] *Crossref*. URL: `https://www.crossref.org/blog/python-and-ruby-libraries-for-accessing-the-crossref-api/`.

[9] Bin-Ge Cui and Xin Chen. "An improved hidden Markov model for literature metadata extraction". In: *Advanced Intelligent Computing Theories and Applications* (2010), pp. 205–212.

[10] Pradeep Dasigi et al. "Experiment Segmentation in Scientific Discourse as Clause-level Structured Prediction using Recurrent Neural Networks". In: *arXiv preprint arXiv:1702.05398* (2017).

[11] Barbara Gastel and Robert A Day. *How to write and publish a scientific paper*. ABC-CLIO, 2016.

[12] Giovanni Giuffrida, Eddie C Shek, and Jihoon Yang. "Knowledge-based metadata extraction from PostScript files". In: *Proceedings of the fifth ACM conference on Digital libraries*. ACM. 2000, pp. 77–84.

[13] Mark Hall et al. "The WEKA data mining software: an update". In: *ACM SIGKDD explorations newsletter* 11.1 (2009), pp. 10–18.

[14] Hui Han et al. "Automatic document metadata extraction using support vector machines". In: *Digital Libraries, 2003. Proceedings. 2003 Joint Conference on*. IEEE. 2003, pp. 37–48.

[15] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. "Overview of supervised learning". In: *The elements of statistical learning*. Springer, 2009, pp. 9–41.

[16] Peter Hurley et al. "Introduction to PostScript". In: *A Sun User's Guide*. Springer, 1992, pp. 84–106.

[17] Roman Kern et al. "TeamBeam-meta-data extraction from scientific literature". In: *D-Lib Magazine* 18.7/8 (2012).

[18] Aleksandar Kovačević et al. "Automatic extraction of metadata from scientific publications for CRIS systems". In: *Program* 45.4 (2011), pp. 376–396.

[19] Ryan TK Lin et al. "Using conditional random fields for result identification in biomedical abstracts". In: *Integrated Computer-Aided Engineering* 16.4 (2009), pp. 339–352.

[20] Patrice Lopez. "GROBID: Combining automatic bibliographic data recognition and term extraction for scholarship publications". In: *Research and Advanced Technology for Digital Libraries* (2009), pp. 473–474.

[21] Xiaonan Lu et al. "A metadata generation system for scanned scientific volumes". In: *Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries*. ACM. 2008, pp. 167–176.

[22] Minh-Thang Luong, Thuy Dung Nguyen, and Min-Yen Kan. "Logical structure recovery in scholarly articles with rich document features". In: *Multimedia Storage and Retrieval Innovations for Digital Library Systems* 270 (2012), p. 2.

[23] Christopher D Manning, Hinrich Schütze, et al. *Foundations of statistical natural language processing*. Vol. 999. MIT Press, 1999.

[24] Simone Marinai. "Metadata extraction from PDF papers for digital library ingest". In: *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on*. IEEE. 2009, pp. 251–255.

[25] IC Mogotsi. *Christopher d. manning, prabhakar raghavan, and hinrich schütze: Introduction to information retrieval*. 2010.

[26] Colin Neville. *The complete guide to referencing and avoiding plagiarism*. McGraw-Hill Education (UK), 2010.

[27] *Pdf-extract*. URL: `http://labs.crossref.org/pdfextract/`.

[28] Jacob Perkins. *Python text processing with NLTK 2.0 cookbook*. Packt Publishing Ltd, 2010.

[29] *Scikitlearn*. URL: `http://scikit-learn.org/`.

[30] Pontus Stenetorp et al. "BRAT: a web-based tool for NLP-assisted text annotation". In: *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics. 2012, pp. 102–107.

[31] Johan AK Suykens and Joos Vandewalle. "Least squares support vector machine classifiers". In: *Neural processing letters* 9.3 (1999), pp. 293–300.

[32] Ah-Hwee Tan et al. "Text mining: The state of the art and the challenges". In: *Proceedings of the PAKDD 1999 Workshop on Knowledge Disocovery from Advanced Databases*. Vol. 8. sn. 1999, pp. 65–70.

[33] Ken Thompson. "Programming techniques: Regular expression search algorithm". In: *Communications of the ACM* 11.6 (1968), pp. 419–422.

[34] Dominika Tkaczyk et al. "CERMINE: automatic extraction of structured metadata from scientific literature". In: *International Journal on Document Analysis and Recognition (IJDAR)* 18.4 (2015), pp. 317–335.

[35] Melanie Weis, Felix Naumann, and Franziska Brosy. "A duplicate detection benchmark for XML (and relational) data". In: *Proc. of Workshop on Information Quality for Information Systems (IQIS)*. 2006.

[36] Xiaoli Zhang et al. “A structural SVM approach for reference parsing”. In: *BMC bioinformatics* 12.3 (2011), S7.

[37] Jie Zou, Daniel Le, and George R Thoma. “Locating and parsing bibliographic references in HTML medical articles”. In: *International journal on document analysis and recognition* 13.2 (2010), pp. 107–119.