

SCUOLA  
NORMALE  
SUPERIORE

Class of Science

Ph.D. in Data Science

35° cycle

**Computational Authorship Analysis:  
Applications and Issues in the Cultural Heritage Field**

Scientific Disciplinary Area INF/01

Candidate:

**Dr.ssa Silvia Corbara**

Supervisors:

**Prof.ssa Anna Monreale**

**Dr. Fabrizio Sebastiani**

**Dr. Alejandro Moreo**

---

Academic year 2023/2024



# Abstract

The discipline of Authorship Analysis studies the linguistic style of written documents to determine information about their authorship. Unlike traditional methodologies, it leverages statistical methods and focuses on quantifiable linguistic events rather than the literary content of the text. In recent years, this field has experienced significant growth due to advances in information technology, enabling the employment of Machine Learning and Natural Language Processing computational tools, and it has been applied in various domains, spanning from cybersecurity to forensics.

This Ph.D. Thesis investigates the application of Computational Authorship Analysis methodologies in the cultural heritage domain. Building on the experience gathered through the research of a case-study (the debated Dantean authorship of the historic document *Epistle to Cangrande*), we address what we believe are the four main issues in this domain application: i) the identification of features that allow for accurate classification while being topic-agnostic; ii) the limited size of the datasets usually available in these studies; iii) the challenges that can be encountered when facing the possibility that the document under scrutiny is a forgery; and iv) the necessity of providing scholars in cultural heritage with proper explanations regarding the computational system's findings.

Each of these issues is covered by a dedicated chapter in this dissertation, in which we offer a deep examination of the problem background, describe our proposed solutions, and present the related results of our research on the matter. In particular, we: i) introduce the use of rhythmic features; ii) evaluate the employment of an alternative vectorial representation of the documents, based on the concept of document pairs; iii) propose the augmentation of the classifier training data with automatically generated samples that mimic the work of a forger; and iv) assess the suitability of some modern explainability methods for the cultural heritage public.

With this work, we aim to offer a comprehensive overview of the Authorship Analysis field, and provide guidance on the best practices for its application in cultural heritage.



# Contents

<b>1</b>	<b>Introduction</b>	<b>9</b>
1.1	Task definitions . . . . .	10
1.2	Practical applications . . . . .	12
1.2.1	Cybersecurity . . . . .	12
1.2.2	Forensics . . . . .	13
1.2.3	Cultural heritage . . . . .	13
1.3	Thesis objectives and contributions . . . . .	15
<b>2</b>	<b>Survey of the state of the art on Authorship Analysis</b>	<b>17</b>
2.1	Feature design . . . . .	18
2.2	Similarity-based techniques . . . . .	20
2.3	Classic Machine Learning techniques . . . . .	22
2.4	Deep-Learning techniques . . . . .	24
<b>3</b>	<b>The <i>Epistle to Cangrande</i></b>	<b>27</b>
3.1	Related work . . . . .	27
3.2	The history of the <i>Epistle</i> . . . . .	29
3.3	Experimental setting . . . . .	30
3.3.1	MEDLATIN: Two datasets for medieval Latin . . . . .	30
3.3.2	MedieValla, a tool for Latin AV . . . . .	30
3.3.3	Experimental protocol . . . . .	32
3.4	Results . . . . .	33
3.4.1	Is Dante the real author of the <i>Epistle to Cangrande</i> ? . . . . .	34
3.4.2	Addendum: The <i>Epistle to Henry VII</i> . . . . .	35
3.5	Discussion . . . . .	35
<b>4</b>	<b>New topic-agnostic features for Authorship Analysis</b>	<b>37</b>
4.1	Rhythmic features for Latin . . . . .	37
4.1.1	Related work . . . . .	38
4.1.2	A brief introduction to syllabic quantity, and how to extract it . . . . .	38
4.1.3	Experimental setting . . . . .	41
4.1.3.1	Datasets . . . . .	41
4.1.3.2	Topic-agnostic features: Base features and distorted views . . . . .	42
4.1.3.3	Experimental protocol . . . . .	44
4.1.4	Results . . . . .	48
4.2	Rhythmic and psycholinguistic features for Spanish . . . . .	50
4.2.1	Related work . . . . .	50
4.2.2	Syllabic stress and psycholinguistics for authorship studies . . . . .	51

4.2.3	Experimental setting . . . . .	52
4.2.3.1	Dataset . . . . .	52
4.2.3.2	Topic-agnostic features: Base features and text encodings . . . . .	53
4.2.3.3	Experimental protocol . . . . .	54
4.2.4	Results . . . . .	55
4.2.5	Post-hoc analysis of the AV results . . . . .	57
4.2.5.1	One-way ANOVA test for political groups . . . . .	58
4.2.5.2	Spearman coefficient applied to style indices . . . . .	58
4.3	Discussion . . . . .	62
<b>5</b>	<b>Diff-vectors as an alternative representation for limited-sized datasets</b>	<b>65</b>
5.1	Related work . . . . .	66
5.2	Diff-Vectors: Characteristics and advantages in AId applications . . . . .	67
5.2.1	More training examples for AV settings . . . . .	68
5.2.2	More robust training in AA settings . . . . .	70
5.3	Methodology for solving AId tasks with Diff-Vectors . . . . .	71
5.3.1	Solving SAV . . . . .	71
5.3.2	Solving AA . . . . .	72
5.3.2.1	LazyAA . . . . .	72
5.3.2.2	StackedAA . . . . .	73
5.3.3	Solving AV . . . . .	74
5.4	Experimental setting . . . . .	75
5.4.1	Datasets . . . . .	75
5.4.2	Learners and features . . . . .	76
5.4.3	Experimental protocol . . . . .	77
5.5	Results . . . . .	78
5.5.1	Intrinsic evaluation of Diff-Vectors . . . . .	78
5.5.1.1	Experiments for closed-set SAV . . . . .	78
5.5.1.2	Experiments for open-set SAV . . . . .	82
5.5.2	Extrinsic evaluation of Diff-Vectors . . . . .	85
5.5.2.1	Experiments for AA . . . . .	85
5.5.2.2	Results for AV . . . . .	86
5.5.3	Efficiency . . . . .	87
5.5.3.1	Efficiency analysis . . . . .	88
5.5.3.2	Timings . . . . .	89
5.5.3.3	Are the costs incurred by Diff-Vectors tolerable? . . . . .	90
5.5.4	Can we use Diff-Vectors for “natively binary” AV problems? . . . . .	91
5.5.5	Results with a different learner . . . . .	92
5.6	Discussion . . . . .	93
<b>6</b>	<b>An attempt to unveil authorship forgery via data augmentation</b>	<b>95</b>
6.1	Related work . . . . .	96
6.2	Methodology for synthesizing forged documents . . . . .	97
6.2.1	Generator architectures . . . . .	97
6.2.2	Generator training . . . . .	99
6.3	Experimental setting . . . . .	100
6.3.1	Datasets . . . . .	100

6.3.2	Learners and features . . . . .	101
6.3.3	Experimental protocol . . . . .	102
6.4	Results . . . . .	103
6.4.1	Possible explanations of negative results . . . . .	106
6.5	Discussion . . . . .	109
<b>7</b>	<b>EXplainable Authorship Analysis: Tools for a new perspective</b>	<b>111</b>
7.1	Related work . . . . .	112
7.2	Tools for eXplainable Authorship Analysis . . . . .	114
7.2.1	Feature ranking . . . . .	116
7.2.2	Probing . . . . .	116
7.2.3	Selection of factuials and counterfactuals . . . . .	118
7.3	Experimental setting . . . . .	118
7.3.1	Dataset . . . . .	118
7.3.2	Learning methods . . . . .	119
7.4	Comparative analysis for eXplainable Authorship Analysis in cultural heritage . . . . .	119
7.4.1	Feature ranking for SAV . . . . .	120
7.4.2	Probing the Transformer for AA . . . . .	122
7.4.3	Factuials and counterfactuals for AV . . . . .	124
7.5	Discussion . . . . .	126
<b>8</b>	<b>Conclusions</b>	<b>129</b>
	<b>Appendices</b>	<b>131</b>
	Appendix A : Formulas . . . . .	131
	A.1 Evaluation metrics . . . . .	131
	A.2 Chi-square . . . . .	131
	A.3 TfIdf . . . . .	132
	Appendix B : Datasets . . . . .	132
	B.1 MEDLATIN: MEDLATINEPI and MEDLATINLIT . . . . .	133
	B.2 LATINITASANTIQUA . . . . .	135
	B.3 KABALACORPUSA . . . . .	137
	B.4 PARLAMINT . . . . .	137
	B.5 IMDB62 . . . . .	137
	B.6 PAN11 . . . . .	138
	B.7 VICTORIAN . . . . .	138
	B.8 ARXIV . . . . .	138
	B.9 TWEETFAKE . . . . .	138
	B.10 EBG . . . . .	139
	B.11 RJ . . . . .	139
	Appendix C : Latin function words . . . . .	139
	Appendix D : Implementation details . . . . .	139
	<b>Bibliography</b>	<b>141</b>
	Papers published during this Ph.D. course . . . . .	141
	Primary sources . . . . .	142
	Secondary sources . . . . .	162



# Chapter 1

## Introduction

The investigation of the authorship of written documents has been a methodological and practical problem as long as written documents have existed. Even in ancient times, scholars quarreled on the genuine or spurious nature of salient literary works, and whether it was appropriate to include or exclude them from the canons of their supposed authors. For example, Terentius Varro (1st century BCE) compiled the *De comoediis Plautinis*, where he divided the 130 comedies attributed at the time to the comedian Titus Maccius Plautus in authentic, dubious, and spurious; his work had such reputation, that only the 21 comedies that he regarded as authentic have been copied and transmitted to us, while the others are now lost. Conversely, the famous *Donation of Constantine*, where the emperor Constantine the Great allegedly transferred the authority over Rome and the western part of the Roman Empire to the Pope, was repeatedly used by the papacy as a powerful argument to justify the temporal power of the popes and the interventions against the secular powers of the West, at least until Lorenzo Valla (15th century) proved unequivocally that the donation was a forgery [90, 156, p. 100].<sup>1</sup> However, the study of the authorship of written texts is not limited to texts from a remote past, without any applicability to the modern world. Only some decades ago, the terrorist known as Unabomber was sadly able to kill three people and injure 23 others in a terrible mail bombing campaign, before the FBI was able to identify him as Theodore John Kaczynski. This conclusion was reached also thanks to the linguistic similarities found between the felon’s anti-industrialization manifesto *Industrial Society and Its Future*, and his personal written production (the correspondence with his brother, plus some academic essays) [180, 184]. It is thus clear that the research on the authorship of written texts has a major influence in many legal, social, and cultural aspects of our society and history.

In general terms, this discipline, known as *Authorship Analysis (AA)*, can be defined “broadly as any attempt to infer the characteristics of the creator of a piece of linguistic data” [152, p. 238]; this definition thus includes not only the author’s true identity, but also their biographical information (e.g., age, gender, mother tongue, etc.). In its long history, scholars have mostly relied on traditional methods provided by disciplines such as philology, paleography, history, and forensics; for example, the authorship of a document can be inferred by the historical events it mentions, or by the characteristics of the physical medium on which the text is written, or by the analysis of the handwriting. However, in the late 19th century, scholars began to experiment with some new methods, based on the application of statistical tools; Wincenty Lutosławski [195] was the first to employ the term *stylometry* to define this novelty. The main difference between these methodologies and the traditional ones is in the focus: researchers do not rely on the investigation of the artistic value, or on the meaning of the written work, or on

---

<sup>1</sup>For a detailed survey of similar literary disputes, however limited to classic and English literature, see Love [192, pp. 14–31]

exogenous evidence such as the handwriting, but on a quantifiable characterisation of the author’s style. This characterisation is typically achieved through the analysis of the frequencies of occurrence of certain linguistic events, such as words, punctuation marks and syntactic constructs. Indeed, it is assumed that there is a set of linguistic events (known as *style markers*) that are related to the communication habits of an author (*idiolect*), or group of authors that share some common characteristic (epoch, literary genre, native language, etc.). In this view, each author can thus be identified by a unique *stylistic fingerprint*, or “human stylome”, that remains more or less constant throughout the author’s production, while conversely varying substantially across different writers, and that is not the result of some influence by others’ works [90, 152, 290]. These are somewhat strong assumptions, since it is realistically preposterous to affirm that an author was not affected by the production of colleagues and predecessors to at least some degree, and since it has been proven that a person’s writing style does indeed change through time and life events [26], and depending on the intended recipient of the text [139]. Still, there are good reasons to believe that such authorial fingerprint, although complex to define, indeed exists, given the unique way each person experiences and learns a language; this hypothesis is also confirmed by the experimental results obtained in many recent computational AAn experiments [162, 163].

Moreover, in recent years the increased availability of computational power and storage capacity, as well as the technological advances in many areas of information technology, has strongly helped AAn, making easily accessible many digitised datasets and new automatic tools for statistical computation; in particular, the field of Artificial Intelligence (AI), and especially its sub-fields of Machine Learning (ML) and Natural Language Processing (NLP), have had an extraordinary impact on the AAn field. In fact, nowadays AAn methodologies based on stylometry are implemented exclusively as computational algorithms.<sup>2</sup>

This trend can also be seen in the bibliometric analysis by Michailidis [203]: not only the research around stylometry has had an annual scientific production growth rate of 13.5% in the period 1968–2021, and was especially strong in the last decade (2011–2021), but the term “machine learning” has also appeared among the most frequent keywords in this research since the year 2006.

While AAn has practical applications in many fields, spanning from cybersecurity to forensics, it is especially important for the cultural heritage domain. Indeed, the paternity of a vast number of literary works is nowadays unknown or uncertain, usually due to the difficult transmission of the writing through the centuries. Here, AAn methodologies can be a powerful aid for scholars: many considerations and points of view that were concealed before can be revealed through stylometric analysis, which would be unfeasible if carried out manually. However, AAn as applied to the cultural heritage field faces specific issues that have to be carefully tackled in order to carry out a meaningful investigation.

In this Ph.D. Thesis, we focus on the applications of AAn to the cultural heritage domain. In the next paragraphs, we introduce the subject matter of the Thesis: in Section 1.1 we formally define the various AAn tasks, in Section 1.2 we discuss the most important applications of the AAn methods, and finally in Section 1.3 we outline the objectives of the Thesis, describing the research questions and the solutions proposed.

## 1.1 Task definitions

AAn branches into different groups of tasks, depending on the specific goal of the study. These tasks are usually approached via automatic *text classification*, whose goal is to assign a document to one or more classes within a predefined set [15, 259]. In automatic classification, *multi-class* (as opposed to *binary*)

---

<sup>2</sup>From here on, we employ the term “authorship analysis” and “stylometry” meaning specifically “computational authorship analysis” and “computational stylometry”, unless otherwise specified.

means that there is a set of  $m > 2$  possible classes (or labels) to choose from; there are instead only 2 classes to choose from in the *binary* case. On the other hand, *multi-label* (as opposed to *single-label*) means that zero, one, or more than one class may be attributed to each item; exactly one class must instead be attributed to any given item in the *single-label* case.

Two of the main groups of tasks of AAn are *Authorship Identification* and *Author Profiling*. The goal of **Authorship Identification (AId)** is to investigate the true identity of the author of a written document when it is unknown or debated. Here, given a finite set  $\mathcal{A}$  of authors and a domain  $\mathcal{D}$  of documents, for each document  $d_i \in \mathcal{D}$  we denote by  $y_i \in \mathcal{A}$  the true author of  $d_i$ . We assume the existence of a training set  $\mathcal{L} = \{(d_i, y_i)\}_{i=1}^n$  of documents of known paternity. AId can be divided into three main tasks:

- **Authorship Attribution (AA)**: given a document  $d_i$  and  $m$  candidate authors  $\mathcal{A} = \{A_1, \dots, A_m\}$ , the goal is to identify the real author of  $d_i$  among the set of candidates [174, 268]. AA is thus a single-label multi-class classification problem, where the classes are the authors in  $\mathcal{A}$  [269]. If it is guaranteed that the real author of  $d_i$  is in  $\mathcal{A}$ , the task is called *closed-set AA*, and is called *open-set AA* otherwise; the latter is understandably a more difficult setting [152].
- **Authorship Verification (AV)**: given a document  $d_i$  and a candidate author  $A^*$ , the goal is to find whether  $A^*$  is the author of  $d_i$  or not [269]. AV is often considered in the literature a one-class classification problem where a text is either classified as part of the given class (the author  $A^*$ ), or as a negative example of the class [122, 175, 235, 275]. Nevertheless, it is more commonly considered a binary classification problem, with  $\mathcal{A} = \{A^*, \bar{A}^*\}$  as the possible classes, by collecting an appropriate number of samples from authors that are not  $A^*$  [234]; as such, in this Ph.D. Thesis, we adhere to this latter formulation. Note that, given the nearly infinite number of authors that are not the author of interest  $A^*$ , selecting a representative sample of authors and documents is far from trivial [121, 175].
- **Same-Authorship Verification (SAV)**: given two documents  $d_i$  and  $d_j$ , the goal is to infer whether they are written by the same author or not, regardless of who the author is; SAV is thus a binary classification problem, with **Same** and **Different** as the possible classes. For this task, knowing the exact identity of the author(s) of each pair is not essential. However, if the labels of the individual documents within the pairs are known, this task admits two different variants: closed-set SAV, where the authors of the documents are assumed to be in  $\mathcal{A}$ , and open-set SAV, where the authors of the documents are not necessarily in  $\mathcal{A}$ .

Some authors [121, 162] consider SAV a special case of AV, where the available set of textual samples by  $A^*$  counts as only one document. In order to avoid ambiguity [269], we prefer to maintain two distinct definitions, since this allows to clearly separate the task of predicting whether a document  $d_i$  is by a candidate author  $A^*$ , from the task of predicting whether a document  $d_i$  is by the same author as some other document. This also allows more general definitions, since SAV as defined here does not assume the author of one of the two documents to be known. Some authors [20, 176, 234] have also noted that AV can be considered a special case of open-set AA with a single candidate author. Koppel and Winter [176] note that SAV can be considered the “fundamental problem” of AId, since an AA problem can be recast as a series of AV problems (where the output would be positive for the true author and negative for the other candidates), with the assumption of the system returning a single positive AV decision.<sup>3</sup>

---

<sup>3</sup>Note that this assumption cannot be guaranteed: there could always be more than one authorship verifier returning a positive decision, or none at all. What can be done is to employ partial AV decisions (e.g., classification probabilities) and aggregate them in a way that ensures a single final AA decision.

Likewise, an AV problem can be recast as a series of SAV problems (where the SAV pair would be made of the document in question and a document by  $A^*$ , and the output would be positive if  $A^*$  was also the author of the document in question, negative otherwise), with the assumption of having documents written by  $A^*$  available, which is not required in standard SAV cases.<sup>4</sup>

AId is closely related to the task of *plagiarism detection* and *style change detection*: the former aims at spotting the illicit use of another person’s ideas and/or linguistic production, without proper acknowledgment of the original source [274], while the latter aims at defining whether a document has a single author or multiple authors [168].

The goal of **Author Profiling (AP)** is to distinguish among classes of authors, in order to predict the authors’ individual characteristics, which can range from their gender [240], to their native language [35], to their nationality [228], to their mental state [127]; these studies are especially important in order to find common traits among groups, or to reveal relevant information about a specific author. As such, the formulation of the specific AP task depends on the type and number of the classes considered.

## 1.2 Practical applications

In general, computational algorithms offer a series of advantages over human-based analysis: they are undoubtedly cheaper in terms of resources (especially time), making the processing of huge amounts of data quickly accessible; something that would otherwise be unfeasible. Their processes are also easily reproducible, allowing for a more accurate and comprehensive evaluation of the proposed methods [155]. AAn techniques can thus be applied in a variety of different fields, where they can be a powerful research aid [154].

In the next paragraphs, we will offer a brief overview on the three main applications of AAn: cybersecurity (Section 1.2.1), forensics (Section 1.2.2),<sup>5</sup> and cultural heritage (Section 1.2.3).

### 1.2.1 Cybersecurity

The digital era allows users to maintain various levels of anonymity, either with the freedom to create online accounts with no direct association to the user’s real identity, or with procedures such as masking the real IP address by connecting to a proxy server. This gives way to dishonest and nefarious individuals to perpetrate crimes via digital means, such as phishing, cyber-bullying and identity theft. Cybersecurity is the field that aims to design techniques for preventing exactly such illegal actions.

In this landscape, AAn can be usefully employed to block, discourage, or even track down the authors of such crimes. Many projects focus on tackling AAn tasks specifically for online messages, such as emails, blogs, social media posts, or tweets [10, 83, 139, 257]; these media are particularly difficult to approach, since the communication is often short and poorly cared for, with plenty of slang terminology and with grammar and spelling mistakes. In particular, AAn can aid in monitoring the shared content on social media [244], and avoiding posts that might go against the terms of service of the media, and generally against the law. A similar task is the detection of aliases, i.e., linking multiple online accounts to known individuals [13, 221]; to this end, external information can also be used, such as the time distribution of the messages or the social network of the profiles [24, 147].

AAn applied to online messages might also help addressing the widespread phenomenon of the diffusion of fake news [237] and the creation of fake reviews [181], that might unjustly impact a business (positively

---

<sup>4</sup>The same reasoning as in Footnote 3 applies here. This is also one of the topics of Chapter 5.

<sup>5</sup>Note that there is a certain overlap among the areas of cybersecurity and forensics (with digital evidence). For the sake of the discourse, we intend the former as mostly regarding the prevention of illegal threats in the online sphere, and the latter as the investigation of crime already committed, with on-going trials.

or negatively) and deceive potential customers; contrasting these misconducts would conversely encourage the sharing of real and official information. Moreover, given the extraordinary advances seen in the text generation field, being able to detect texts written by automatic algorithms (which usually hide malicious intents, such as spamming or review-bombing) is becoming a critical necessity [95, 248, 287].

Finally, there is also a specialized branch of AId that focuses on the identification of software developers, where the written documents are files or pieces of source code [50, 82]. Although programming languages are typically restrictive in terms of syntax and grammar, they still allow enough flexibility to have discriminative recurring patterns and preferences among equivalent solutions [170]. This research can help preventing the propagation of potentially dangerous viruses and malware, in addition to solving copyright issues [102].

### 1.2.2 Forensics

Written texts can be crucial evidence in the investigation of crimes. It goes without saying that also the ability to precisely identify, or at least characterize, the author of such written texts can have a decisive importance in forensic cases (an example is the Unabomber case already presented in Section 1).

In the past, the employment of AAn as actual proof in court trials was seen with distrust, mostly due to the insufficient level of scientific validation of the AAn methods [128, 265]. Indeed, there is still a vivid dispute on the matter [77, 265] and, generally speaking, it seems that AAn research is considered more suitable for leading the investigation process by narrowing down the potential pool of suspects [228, 244].

However, while depending on the local regulations, juries and courts usually recognize the validity of the linguistic analysis, and allow experts to offer testimony, as long as the accuracy, consistency, and reliability requirements of the methods are adequately reported [152, 155]. Basically, AAn can be fruitfully used in every criminal case where a written document is involved, from newspapers articles for immigration issues [153], to emails for racial discrimination [70], to threat letters [184] or suicidal notes [70] for murder cases. In a historical case, Nini [216] even put the letters by the infamous murderer “Jack the Ripper” under scrutiny, strengthening the hypothesis that the author of the two most renowned letters is indeed the same person.

### 1.2.3 Cultural heritage

As already mentioned, AAn finds a natural use in the cultural heritage field, where scholars (philologists, historians, etc.) can profit from the analysis and new perspectives brought to bear by AAn statistical techniques. In fact, it could be said that one of the first use of modern AAn methodologies is indeed in the cultural heritage domain: it is the pioneering work of Mosteller and Wallace [208], who applied multivariate statistical analysis to the *Federalist Papers*, a collection of 85 articles and essays written in support of the Constitution of the United States by Alexander Hamilton, James Madison and John Jay, but under the collective pseudonym “Publius”, thus creating an “authorial mystery” regarding the individual contributions to each document (see also Section 2.1).

In the literature, most AAn studies regarding cultural heritage investigate the AId problem of documents with debated or unknown paternity. This is especially the case for ancient texts: with many centuries of transmission, the history of the writing can easily become lost or dampened, thus sparking doubts and subsequent debate regarding its true authorship. In particular, many works in the field concern Latin authors and collections, from renowned authors like Julius Caesar [167], Pliny the Younger [284] or Cicero [288], to saints [165], to even unnamed individuals [156]; instead, in the English language, case studies related to Shakespeare are very popular [53, 230, 279]. On the other hand, many modern AId works attempt to disclose the identity of authors who originally wanted to remain anonymous [133, 199], sometimes even though (or because?) they were already famous [32, 154], or as a possible editorial

strategy [43, 130]. However, AId can also be used in order to underline the stylistic homogeneity of a literary work such as *Beowulf* against the hypothesis of composite authorship [212], or on documents whose authorship is not disputed, in order to analyze the influence a certain author might have had on another [101].

Some projects tackle the AP task as well, either to comparatively analyze the identity of a mysterious author such as Elena Ferrante [286], or to reveal characteristics of undisputed authors, such as mental illness [127].

AAn in cultural heritage suffers from specific domain issues [266], most importantly the extremely limited number of textual samples available. It could be argued that, even in contemporary applications, gathering enough textual material for a meaningful authorship research is not a trivial matter, since a single person cannot conceivably produce more than a moderate amount of written texts in their lifetime. However, nowadays it is fairly easy to rapidly and effortlessly harvest hundreds or thousands of documents through digital sources such as Twitter, blogs and the like, a type of resource that also grows exponentially every day. On the contrary, this is not a viable option for many cultural heritage studies, since the authors under scrutiny usually have already passed away. This situation is even worsened for AAn cases that deal with dead languages, where even the documentation more generally written in the language is scarce, and it is unlikely to increase in the future (some new documents might be discovered, but rarely in a significant amount). This is a strong limitation for AAn works on cultural heritage, especially since it often prevents the employment of complex AI methods that would require massive datasets; indeed, Luyckx and Daelemans [196] underline how many AAn works report results obtained on data that are unrealistic in real-world settings. Furthermore, many languages (dead, or used by limited groups) face a worrying gap in the availability of specifically developed NLP tools.

Notably, cultural heritage professionals tend to look at AAn methodologies with strong reservations, if not with utmost distrust. This is usually caused by the false impression that AAn experts presumptuously consider the authorship classification results to have the same authoritativeness as the analysis of the human expert. Quite the contrary, it is important to underline that AAn techniques are by no means to substitute traditional methods used by domain professionals, nor AAn researchers should aim to do so, and the classification results should be considered with the right degree of caution [130]. The output of an AAn research should be merely used, as already claimed, to complement the human efforts of the expert as helpful investigation tool [128, 246], not unlike other AI methods used, for example, to help medical staff in a diagnosis.

Moreover, note that the domain expert and the AAn system often employ complementary methodologies. For instance, a domain expert may (i) analyse the historical facts described in the text and check whether a certain candidate author could possibly have been aware of these facts; (ii) analyse the stand that a candidate author takes towards a certain issue, and check whether this stand is compatible with what we already know about the author’s ideas; and (iii) in general, bring to bear our collective knowledge of a given candidate author, of the historical period in which they operated, of the cultural *milieu* that surrounded them, and decide whether all these are or not compatible with the hypothesis that they may be the real author of the disputed document. Current AAn systems can do none of the above. More in general, while the domain expert can use *exogenous* real-world knowledge, an AAn system is typically only able to use *endogenous* knowledge (plus potentially some external linguistic knowledge, in the form of dictionaries, or sets of word embeddings). However, an automatic system is capable of doing fine-grained statistical analyses on this endogenous knowledge that would be difficult, or impossible, for any human to perform;<sup>6</sup> stylometric analysis is indeed one such type of analysis, where an automatic

---

<sup>6</sup>Domain experts sometimes do analyse the same features as automatic systems, e.g., they may notice that an author tends to use a specific spelling of a given word, or that an author often starts a sentence with a certain word or sequence

system can compute a huge amount of frequencies (of events of apparently minimal significance) that are ultimately correlated to the author’s style. In other words, this “lower-level” analysis of the text provides a useful complement to the “higher-level” analysis that the domain expert carries out.

### 1.3 Thesis objectives and contributions

As already stated, this Thesis focuses on the application of AAn to the cultural heritage domain. It unfolds from an actual AId case study: the *Epistle to Cangrande*, a document in medieval Latin whose authorship by Dante Alighieri has been a source of debate for centuries. We use this project as a starting point to introduce the main issues that are intrinsic in the application of AAn methodologies, and specifically to the cultural heritage studies. These issues can be summarized into four main problems:

- Problem 1.** An AAn system, even one with apparently good performance, might actually leverage domain-related information, instead of style-related information. In this case, the system would be actually performing topic identification instead of AAn, which is undesirable [45, 121]. Hence, an AAn system should avoid this kind of information, and employ purely stylistic data.
- Problem 2.** As already clarified in the previous section, a specific problem of AAn in the cultural heritage domain is the limited size of the textual datasets that scholars usually have at their disposal. This is an issue of apparently impossible solution in most cases, given that no new original documents can be produced, and it often inhibits the employment of AI/ML techniques in important literary and historical studies. Therefore, there is a critical need for a solution to ameliorate the limitations of dataset size.
- Problem 3.** Developing an effective AId system is extremely difficult if an “adversary” is at play, i.e., a process (or a human) that actively tries to fool the system [54, 236]. Our cultural heritage is indeed filled with countless examples of historical documents of questioned authorship, often caused by supposed forgeries or false appropriations [76, 199, 216, 256, 284, 288]. An AId system should therefore be robust to this form of adversarial attack.
- Problem 4.** In cultural heritage, the domain specialists (philologists, historians, etc.) are professionals who devoted (part of) their studies to the investigation of a certain authorship problem. As such, they give no actual value to an opaque classification output; on the contrary, they find value in the reasons for the classification output, both to validate the classification process itself, and to gain new insight on the problem that can then be further explored. Thus, an AAn system should be equipped with some form of explanatory ability.

In the next chapters, we present various methods and analyses to address these aforementioned issues. In particular, this Thesis is organized as follows:

- In Chapter 2 we offer an overview of the state of the art of the AAn field, presenting the relevant literature.
- In Chapter 3 we present the study regarding the *Epistle to Cangrande*, contextualizing the major issues that are typical of AAn applied to the cultural heritage domain. The research presented in

---

of words. However, it is undeniable that a human carries out this type of analyses only with greater difficulty, and can do this only on a limited scale.

this chapter is published in three articles by Corbara et al. [7, 8, 76], and is based on the preliminary work conducted in the MSc Thesis by Corbara [75].

- In Chapter 4, referring to Problem 1, we propose the use of topic-agnostic features based on rhythmic traits, and assess the extent to which this kind of features convey useful information to define authorial characteristics in Latin and Spanish. The research presented in this chapter is published in three article by Corbara et al. [1, 2, 6].
- In Chapter 5, referring to Problem 2, we analyze the application of an alternative method for generating vectorial representations of textual documents for AId, where a vector does not represent a single document, but a pair of documents, arguing that it would allow to better exploit the information within the dataset. The research presented in this chapter is published in an article by Corbara et al. [5].
- In Chapter 6, referring to Problem 3, we propose to tackle the adversarial problem by augmenting the classifier training data with synthetic samples created by automatic generators. The research of this chapter is published in an extended abstract and an article by Corbara and Moreo [3, 4].
- In Chapter 7, referring to Problem 4, we survey some modern methods for producing explanations for ML techniques, evaluate their suitability for AId tasks in the cultural heritage setting, and demonstrate their practical application. The research of this chapter is published in an article by Setzu et al. [9].
- In Chapter 8, we offer some final conclusions.

## Chapter 2

# Survey of the state of the art on Authorship Analysis

As mentioned in Section 1.1, nowadays AAn tasks are tackled as text classification problems [259], where a text is classified as belonging to one or more classes/labels. In turn, text classification is usually solved via ML methods, that enable a computer system to make predictions or take decisions using past data or experiences, without being explicitly programmed. In this regard, the annual PAN shared tasks offer a very good overview of the most recent trends in AAn (see for example the last editions [36, 37, 38, 39]).

We can divide AAn techniques in two groups: (a) **similarity-based** methods, where a certain metric is employed to define the distance among different texts, without a proper learning process, and (b) **ML-based** methods, where a classifier is trained to recognize the statistical characteristics of various classes, and hence to assign a text of dubious authorship to its more probable class. However, in the literature there are other possible categorization to divide AAn methods into:

- **Intrinsic vs Extrinsic:** specific for models that tackle the AV task, this categorization divides the models between those that employ only texts from the author of interest, hence as a one-class classification problem (intrinsic), and those that gather additional texts by other authors, hence as a binary classification problem (extrinsic) [122, 269]; see also Section 1.1. Stamatatos et al. [271] remark that the best-performing submissions in recent PAN events follow the extrinsic paradigm.
- **Profile-based vs Instance-based:** this categorization divides the models between those that concatenate all the available training texts by a certain author in one single document (profile-based), and those that instead process each training text as a separate instance (instance-based). While the former provide a more robust stylometric representation when the length of the texts is limited, the latter is able to better exploit the existing differences among texts [268, 269]. Potha and Stamatatos [234] remark that the majority of existing approaches, and the most successful submissions in recent PAN tasks, follow the instance-based paradigm.
- **Lazy vs Eager:** this is a more general categorization for ML algorithms, which divides the models between those that, during the training phase, simply store the training data (lazy), and those that process the training data in order to extract a general classification model (eager) [269]. Understandably, the former do not incur any computational cost at training time and postpone all of the computational burden to the test phase, and vice versa.

In AAn, documents are usually represented as numerical vectors, in what is known as the Vector Space Model (VSM) [152]: each value in the vector defines the document in a given dimension, known as *feature*.

Specifically, each document  $d_i$  is represented via a labelled vector  $\mathbf{d}_i$  of features, where the value  $\mathbf{d}_i^k$  of the  $k$ -th feature in vector  $\mathbf{d}_i$  usually represents a certain linguistic event or textual trait as it appears in  $d_i$ . This representation allows to directly compare textual documents as vectors in an  $n$ -dimensional space.

Needless to say, the choice of the features to employ is paramount for the system performance. Usually, word-based features adhere to the Bag-of-Words paradigm, where the presence and position of a certain word is assumed to be independent from the presence and position of the other words [152]; this assumption typically extends to other feature types used to represent documents in the vector space, treating linguistic phenomena as independent of each other. This is a practical approximation in order to process natural language, although it is a misleading and simplistic assumption regarding language *per se*.

There are however alternatives to the VSM approach. For example, a popular method is Latent Dirichlet Allocation (LDA) [47], where each document is represented as a mixture over a set of (latent) “topics”, and each topic is defined as a distribution over words [17, 234, 262]. Moreover, in recent years the field of AAn has seen the progress of Deep-Learning (DL) algorithms,<sup>1</sup> which are artificial Neural Networks (NNs) with multiple hidden layers that are notoriously able to discover the features to exploit, thus freeing researchers from the duty to manually select them; however, these features are “latent”, in the sense that they represent hidden variables, and do not correspond to directly observable events in the text. For example, Doc2Vec [182], an extension of the popular Word2Vec [204], represents the entire document into an induced latent space, embedding it into a single vector that somehow captures the “overall meaning” of the text; however, such spaces, typically known as Distributional Semantic Models (DSM), tend to be difficult to control and supervise.

These representations seek to establish a metaphorical correspondence between document semantics and position in the vector space, and between semantic similarity and distance in the space, akin to the VSM model. However, there exist alternatives to this approach: for example, textual documents can be compared based on the amount of information they share, assessed through a compression algorithm [68, 120].

In the next paragraphs, we survey the literature regarding feature design (Section 2.1), and the most relevant AAn methods, divided in: similarity-based methods, that do not employ a learning process (Section 2.2), “classic” ML methods (Section 2.3), and DL methods (Section 2.4).

## 2.1 Feature design

As mentioned, the features employed for the representation of the documents are of primary importance for an AAn system, in order to properly capture the stylistic traits of the authors. Indeed, the research regarding what features are best to apply for this goal spans the entire history of AAn and stylometry: Rudman [246, p. 360] estimates that roughly a thousand of different feature types were proposed by the end of the 20th century, and the research for good stylistic features seems far from ceasing.

Traditionally, the year 1851 is considered the start of stylometry as a practice, when Augustus de Morgan suggested in a letter to a friend that comparing the lengths of the words employed in the text could have been a good approach for tackling questions of authorship, a hypothesis later investigated by Thomas Mendenhall with his Curve of Composition [201]. Almost 50 years later, Udney Yule [302] suggested investigating sentence lengths as a method for determining authorship. In both cases, the working hypothesis was that the use of longer words (sentences) might be a sign of higher education, and

---

<sup>1</sup>For an introduction to the terminology and theory of Neural Networks and Deep Learning, see Goodfellow et al. [107].

could thus be an indicator of the author’s identity. In the same vein, some later projects focused on the use of conspicuous rare words, or a singular comprehensive measurement of vocabulary richness (such as the type-token ratio). These first attempts tended to produce very poor results [167], and nowadays these features are considered unreliable if used alone [268].

A pivotal change in the history of AAn happened in the ’60s with the renowned study by Mosteller and Wallace [208] on the Federalist Papers. The ground-breaking proposal presented in the paper was to employ the frequency of function words.<sup>2</sup> These features offer two major (interlinked) advantages: (i) they appear with very high frequency, offering better stability and scalability to new documents [167]; (ii) since they are so widespread, but do not convey any true meaning by themselves, they tend to pass as elements with no real significance for the discourse, and thus they are assumed to be used by writers in an unconscious manner, making their peculiar patterns hard to be modified or imitated [161, 184].

The research by Mosteller and Wallace [208] prompted a switch from a small set of low-frequency features, to a bigger set of high-frequency features of apparently minimal significance, such as the use of a punctuation symbol or a conjunction, which is still the established methodology in AAn studies [128, 161]; in fact, this study had such an impact, that the same authorship problem was for long used as experimental ground for many AAn methods [68, 129, 146, 255]. Interestingly, in his essay *Clues*, the noted historian Ginzburg [106] describes the emergence of this analytical approach (which he traces back to the late 18th century), that focuses on facts that usually would not attract cognitive attention, in a number of fields of human activity (e.g., studying the mannerism of the fingers and toes while investigating the authorship of paintings); he calls it the “evidential paradigm”.

More recently, in surveying the results of the PAN-2011 Shared Task, Argamon and Juola [20] describe the features that in 2011 were, and have largely remained, standard for the representation of texts in AId tasks. Here, we can see that nowadays function words remains a standard feature type in AAn [161]. Another common feature type employed in AAn (and more generally in Information Retrieval) tasks is word and character  $n$ -grams. In particular, character  $n$ -grams, especially those capturing affixes and punctuation [251], have been proven to have higher discriminative power than most features [110, 174, 252]. A hypothesis for their good results is that they allow to capture “a bit of everything” [161]: lexical and syntactic information, but also patterns of punctuation symbols and white spaces. The optimal number for  $n$  is debated, since higher values entail more contextual information, although at the expense of a dramatic increase of dimensionality [268]; Grieve [110] estimates that the best  $n$ -grams are character bi- and tri-grams, since longer  $n$ -grams might concur in carrying topical information, instead of stylistic information.

Indeed, it can be problematic if a feature feeds the classification algorithm with information about the topic of the document, instead of its stylistic properties. Surely, even though it is understood that writers might have preferred topics in their production, the co-occurrence of topics is not a good basis for an authorial classification decision. Thus, it is usually not advisable to employ features that capture the topic of the documents in common AAn tasks, since this could lead the AAn system to misfocus (see Section 4 for more details on this problem). Character and (especially) word  $n$ -grams are known to suffer from this drawback, as already stated, while features that pertain to syntax more than to content, such as function words [43, 90, 152] or part-of-speech (POS) tags [141], are considered fairly topic-agnostic.

Yet another issue to keep in mind is that different feature types might have different effectiveness on the performance depending on the language under study. For example, features derived from inflectional

---

<sup>2</sup>“Function words” are words with very little or no semantic content, that are used to build the grammatical architecture of the sentence; e.g., articles and prepositions are function words. This term is sometimes confused with the concept of “stop-words”: these are words that are usually filtered out in the pre-processing steps of many Information Retrieval algorithms. Depending on the case, function words might indeed be stop-words, especially for tasks that conveniently exploit the topical information of the documents; as we see here, this is not the case for authorship.

morphology and word order usually perform poorly for the English language, since it is a weakly inflected language with a rather fixed sentence structure, while performing better for languages with a stronger presence of declensions and more structural freedom, such as Greek or Latin [90, 155]; viceversa, function words tend to perform really well for languages that markedly use them to convey functional linguistic information, such as English [161]. As another example, it would be better to use character  $n$ -grams instead of word  $n$ -grams for analysis regarding Ancient Chinese, since the characters constitute the basic unit of this language [278], while Abbasi and Chen [10] state that word lengths could be less discriminating for Arabic texts, since they are distributed over a smaller range. An identical argument can be given regarding the genre and medium of the documents under study. For example, it would be reasonable to exploit the metric patterns for a study regarding poetry [211, 212], while spelling and grammatical errors tend to appear more frequently in emails and social media posts than in published books. More generally, it is fundamental to take into account the linguistic characteristics of the data before selecting the features to employ.

To summarize, the most common features in AAn are divided into five major types in the survey by Stamatatos [268]:

- **Lexical features:** features based on considering text as a sequence of word tokens (e.g., word and sentence lengths, vocabulary richness, word  $n$ -grams, function words, ...);
- **Character features:** features based on considering text as a sequence of characters (e.g., character  $n$ -grams, compression measures, ...);
- **Syntactic features:** features based on syntax (e.g., POS tags, dependency trees, ...);
- **Semantic features:** features based on semantics (e.g., synonyms, semantic dependencies, ...);
- **Application-specific features:** features specifically engineered for the particular application under study (e.g., HTML tags, use of indentation, emojis, ...).

On the one hand, lexical and character features follow the bag-of-words model and are basically language-independent; that is, they require close-to-none expertise on the specific language they are applied to (with the exception of function words). On the other hand, syntactic features are more complex and subtle, but they require robust and accurate NLP tools in order to give meaningful results.<sup>3</sup>

Nowadays, researchers often employ combinations of various feature types, which is known to yield better results [103, 110].

## 2.2 Similarity-based techniques

Two key elements lie at the heart of similarity-based classifiers: i) a similarity (or dissimilarity) measure, which is used to compare the vectors representing the documents; and ii) a decision protocol, which is used to select and assign a class label to the unknown sample. The approaches within this family of methods thus focus on devising improved similarity measures that succeed in capturing the authorial traits better than standard ones (like the Euclidean distance or the cosine similarity), and/or on efficient ways to employ said measurements in order to reach more reliable predictions. Despite their apparent intrinsic simplicity, these techniques have proven to be rather successful in AAn; indeed, some of the first and (arguably) most famous algorithms in AId are similarity-based.

---

<sup>3</sup>Contrary to this idea, Gamon [103] observes that what truly matters for an NLP system used for AAn is its consistency in making errors. This means that even if the system mislabels a linguistic event, the consistent patterns in these errors can still be useful, and the correlation between linguistic features can be leveraged to identify authorship despite inaccuracies.

One of these famous methods is undoubtedly Burrows’ Delta [59], which computes the mean of the absolute differences between the z-scores for a set of  $n$  words (usually the most frequent words, e.g., function words) among two documents. Mathematically, given a comparison corpus  $D$  and a set of  $n$  words, the z-score for word  $w_i$  in document  $d$  can be expressed as:

$$z(w_i, d) = \frac{f(w_i, d) - \mu(w_i, D)}{\sigma(w_i, D)} \quad (2.1)$$

where  $f$  is the frequency of word  $w_i$  in document  $d$ ,  $\mu$  is its mean frequency in the comparison corpus  $D$ , and  $\sigma$  is its standard deviation. Consequently, the Delta measure among the unknown document  $d_u$  and a document  $d_j$  can be expressed as:

$$\text{BD}(d_u, d_j) = \frac{1}{n} \sum_{i=1}^n |z(w_i, d_u) - z(w_i, d_j)| \quad (2.2)$$

(which is equivalent to the Manhattan distance of the z-scores of the word frequencies). In order to classify  $d_u$ , Burrows’ Delta is computed among  $d_u$  and each comparison text in  $D$ ;  $d_u$  is then labelled with the same class as the comparison text with the lowest Burrows’ Delta score. Argamon [19] proposes some improvements of the Burrows’ Delta in order to reconcile some mathematical inconsistency of the standard formulation, but they do not seem to lead to better results in practice [92]. Burrows’ Delta has been vastly used in projects related to cultural heritage [165, 256]; even recently, McCarthy and O’Sullivan [199] employ it for their work regarding the book *Wuthering Heights*.

Another renowned method in the AID field is the Impostors Method (IM) by Koppel and Winter [176]. Given a feature set,  $k$  feature subsets are created by randomly selecting a fixed number of features from the feature set. Similarly to Burrows’ Delta, IM relies on a comparison dataset, from which a number of so-called “impostors” are selected. In particular, given two documents  $d_u$  and  $d_j$ , the algorithm selects a set of impostors  $J$ , i.e., a set of documents that are not written by the author of  $d_j$ ; then, it computes the score  $s(d_u, d_j, J)$ , which is equal to the number of features subsets for which the similarity (computed with a distance metric of choice) among  $d_u$  and  $d_j$  is higher than the similarity among  $d_u$  and each of the documents in  $J$ . The process is repeated by selecting a set of impostors  $U$ , i.e., a set of documents that are not written by the author of  $d_u$ , and computing the score  $s(d_j, d_u, U)$ . If the average ( $s(d_u, d_j, J)$ ,  $s(d_j, d_u, U)$ ) is greater than a given threshold  $\sigma^*$ , the algorithm predicts that the pair  $(d_u, d_j)$  shares the same author. An evident downside of this method is that it might be difficult to guarantee that the texts in  $J$  and  $U$  are not by the same authors who wrote  $d_j$  and  $d_u$ , respectively. Seidman [261] later introduced the General Impostors (GI) method, a modification of IM to support the comparison of  $d_u$  with multiple documents from the same author; GI itself gained various variations [116, 169, 233]. In the cultural heritage domain, IM was used, for example, by Stover et al. [276] to attribute a newly discovered writing to the Latin author Apuleius, and by Kestemont et al. [164] to assess the contributions (“Rezensionen”) of Johann Wolfgang Goethe to the literary magazine *Frankfurter gelehrten Anzeigen*.

Another popular method is the Common  $N$ -Gram (CNG) [160]. An  $n$ -gram profile is computed for each document, where an  $n$ -gram profile is a set of the character  $n$ -grams present in the document (usually the most common ones). Thus, given two documents  $d_u$  and  $d_j$  and their  $n$ -gram profiles  $p(d_u)$  and  $p(d_j)$ , the dissimilarity is given by the formula:

$$\text{CNG}(p(d_u), p(d_j)) = \sum_{g \in (p(d_u) \cup p(d_j))} \left( 2 \cdot \frac{f(g, d_u) - f(g, d_j)}{f(g, d_u) + f(g, d_j)} \right)^2 \quad (2.3)$$

where  $g$  is a character  $n$ -gram from the union of two profiles, and  $f(g, d_i)$  is the normalized frequency of the  $n$ -gram  $g$  in the document  $d_i$ . The unknown document is then assigned to the same class as the

document whose dissimilarity is minimal.

Modifications of this method exist, for example by comparing the unknown document  $d_u$  with the entire profile of an author (i.e., the profile extracted from their concatenated production) [232], or by combining various  $n$ -gram lengths into an ensemble [142]; Layton et al. [181] evaluate a number of variants of the CNG formulation. Similarly, Brocardo et al. [55] create  $n$ -gram profiles for each author and carry out the classification by computing the ratio of shared unique  $n$ -grams with the unknown document; the unknown document is considered to be written by author  $A^*$  if the ratio is above an author-specific threshold. Tuccinardi [284], in his study of Pliny the Younger’s letters to Adrian, employs the “simplified profile intersection”, which is again based on the size of the intersection among the  $n$ -gram profiles of the unknown document and the  $n$ -gram profile of an author’s production.

The employment of off-the-shelf compression algorithms as a basis for evaluating the similarity among documents is extensively used in the AAn field as well. For example, Halvani et al. [120] utilize the Compression-Based Cosine (CBC) to measure the dissimilarity among two documents, and assign the unknown document to the class of the document with which it shares the minimum CBC value, if such value falls below the average CBC value among the unknown document and the other documents in the dataset. Cerra et al. [68] employ the Fast Compression Distance (FCD), where each document is transformed into a dictionary, and the dissimilarity is given by the number of patterns that are found in both dictionaries. Regarding cultural heritage works, Canettieri et al. [61], in his philological study, use the popular LZ77 compressor to obtain the “relative entropy” among two texts. In particular, given the texts  $d_j$  and  $d_u$ ,  $d_u$  is compressed by using sub-sequences derived from  $d_j$ ; these dissimilarity measures are then employed to construct a tree representation of the entire corpus. Basile and Lana [32] employ the same principle to compute a “Gramsci index” in order to identify the articles written anonymously by Antonio Gramsci. Also Benedetto and Degli Esposti [34] employ both the entropic distance and the weighted difference of the  $n$ -gram frequencies to address the problem of the *Diario Postumo* by Eugenio Montale.

Pavlopoulos and Konstantinidou [223] use the perplexity of a statistical language model trained on the Ancient Greek epic poems, the *Iliad* and the *Odyssey*, to identify outliers from other passages of the Homeric canon. In this case, perplexity is used akin to a similarity measure, where the similarity is determined by how much a language model, based on its training data, anticipates the word sequences of the new documents.

As stated, in these learning algorithms the unknown document is typically classified into the same class as its most similar author or document in the dataset,<sup>4</sup> or into the class of a specific author or document if it meets a certain similarity threshold. Another possible decision protocol is to assign the unknown document  $d_u$  to the class of author  $A^*$  if the average similarity among  $d_u$  and each of the texts by  $A^*$  is above the group-average similarity of the texts by  $A^*$  [65].

## 2.3 Classic Machine Learning techniques

Despite the good results often obtained with similarity-based methods, most of the AAn research is conducted exploiting classic ML methodologies.

These techniques are usually divided among supervised and unsupervised methods. In particular, unsupervised methods do not require labelled data, which allows for a significant resource saving, and are usually employed to gain new information regarding the documents under study and their relations, in the form of cluster analysis. For example, Neidorf et al. [212] corroborate the stylistic association

---

<sup>4</sup>Note that this is actually analogous to employing a KNN algorithm with  $k = 1$  (see Section 2.3).

between the poet Cynewulf and the anonymous work *Andreas* via hierarchical agglomerative clustering, and Ochab and Essler [219] explore various unsupervised clustering methods on a dataset of (transcribed) Greek papyri.

However, supervised ML algorithms are often preferred for AAn [244]: in this case, a learning algorithm trains a classifier by exploiting a reference set of already-labelled samples (i.e., the training set), and the classifier is then used for the classification of unseen, unlabelled documents (i.e., the test set). Part of the training set can be excluded and used as validation (i.e., to test the algorithm on known data before the actual testing phase), in order to optimize the hyper-parameters of the model.

In AAn, many classical ML methods have been used, such as K-Nearest Neighbors (KNN) [17] and decision trees [40]; still, arguably the most popular learning algorithms are Support Vector Machine (SVM) and Logistic Regression (LR). In particular, SVM is a standard learning algorithm for many text classification tasks, due to its robustness to high dimensionality and to its general applicability, and is often shown to outperform other learning algorithms such as decision trees and even NNs in various settings [103, 307], especially in regimes of data scarcity [51]. Indeed, SVM has often been the method of choice for both AId tasks, as testified in past PAN editions [166, 168], and AP tasks [80], as well as for cultural heritage studies [60, 100, 101]. On the other hand, a classifier trained via LR has the advantage of directly returning (fairly) well-calibrated posterior probabilities, which can be directly used as a measure of the confidence of the authorship classifier on the classification decision, as noted in Corbara et al. [5, 7], Chhatwal et al. [71].<sup>5</sup> Like SVM, LR has been widely used for both AId [221] and AP [21, 213], and specifically in cultural heritage settings [74, 156].

While these are general-purpose learning algorithms, the Unmasking technique, proposed by Koppel et al. [175], is a popular supervised method specifically designed for AAn. The intuitive idea behind this approach is to iteratively remove those features that are most useful for distinguishing the production of author  $A^*$  and the document  $d_u$ , and to examine the speed at which cross-validation accuracy degrades as more features are removed; if  $A^*$  is also the author of  $d_u$ , then the differences should be limited to a relatively small number of features, thus determining a steeper degradation curve. With a meta-learning scheme, it is then possible to determine which curves represent a “same-author” scenario, as opposed to a “different-author” one. Since in the original formulation  $d_u$  is assumed to be book-length, so that it can be chunked in multiple portions, Bevendorff et al. [42] developed a variation that allows also the processing of shorter texts. In AAn, this method has been used in order to assess whether Alzheimer’s disease would lead to marked changes in authors’ written style [127], as well as to analyze the differences in style between the political left- and right-wing news, and between hyperpartisan and mainstream news [237].

Finally, it has been noted that even simple combination rules typically lead ensemble-based classifiers to outperform any of their base member classifiers [11, 79]; Stamatatos et al. [272] find that a simple

---

<sup>5</sup>The classifiers trained with certain learning algorithms (such as LR) are known to return reasonably well-calibrated posterior probabilities. Those trained with some other learning algorithms (such as Naïve Bayes) return probabilities which are known to be not well calibrated [85], and yet other learners (such as SVMs or AdaBoost) train classifiers that do not output probabilities at all, but instead output confidence scores (which, unlike probabilities, do not range on  $[0, 1]$  and/or do not sum up to 1). In order to address these two latter cases, there are probability calibration mechanisms that convert the outputs of these classifiers into well-calibrated posterior probabilities [214, 215, 229, 297, 303]. An intuition of what “well-calibrated posterior probabilities” means can be provided by the following example. If 10% (resp., 90%) of all the documents  $d_i$  for which a classifier  $h$  outputs  $h(d_i) = \Pr(A^*|d_i) = 0.5$  indeed belong to class  $A^*$ , we can say that the classifier  $h$  has overestimated (resp., underestimated) the probability that these documents belong to  $A^*$ , and that their posteriors are thus inaccurate. Conversely, if this percentage is 50%, we can say that the classifier  $h$  has correctly estimated the probability that these documents belong to  $A^*$ , and that their posteriors are thus accurate. Indeed, we say [99] that the posteriors  $h(d_i) = \Pr(A^*|d_i)$  are perfectly calibrated (i.e., accurate) with respect to a (labelled) set  $\sigma = \{(d_i, y_i)\}_{i=1}^n$  if, for all  $\alpha \in [0, 1]$ , it holds that:

$$\frac{|\{(d_i, y_i) \in \sigma \mid h(d) = \alpha, y_i = A^*\}|}{|\{(d_i, y_i) \in \sigma \mid h(d) = \alpha\}|} = \alpha \quad (2.4)$$

meta-model that averages all the verification scores produced by the methods submitted to the AId task of PAN-2014 is able to perform better than any individual classifier.

## 2.4 Deep-Learning techniques

Despite the good results obtained in other NLP tasks [300], for a long time DL methods have rarely been employed in AAn. Arguably, this is due to the huge quantity of training data they usually require, which is often lacking in AAn settings (see also Section 1.2.3). Indeed, even though DL techniques have made an occasional appearance in some precursor works [279], and even though the winner of PAN-2015 [271] was the NN model by Bagnall [27], until recently it was generally accepted that “simple approaches based on character/word  $n$ -grams and well-known classification algorithms are much more effective in this task than more sophisticated methods based on deep learning” [168, p.9]. Nevertheless, nowadays DL methods are employed more frequently in AAn tasks, again as reported in more recent PAN editions [38, 39].

As mentioned above, contrary to similarity-based and classic ML methods, researchers do not need to pre-define the features to employ with DL techniques. NNs are able to autonomously find correlations in the data, and can work directly with the raw written texts. Indeed, since NNs learn a latent representation of the texts that is used to reach the classification decision, it has been argued that NNs “blur the boundaries between classification-based and similarity-based approaches” [247, p. 1]. However, this peculiarity is also a potential drawback, since the researcher has no control over the features at play, and so the inner workings of the method are harder to grasp (see also Problem 3 in Section 1.3), due to the complexity of DL models.

For AAn, NNs can be used as classifiers, with the final layer returning the classification decision. The most basic schema for a NN architecture is surely a series of dense layers, or multiple parallel series of linear layers, each one processing a different feature type, in a multi-channel approach where the outputs are concatenated before the final decision layer [202]. The second most intuitive choice would be resorting to Recurrent Neural Networks (RNNs). RNNs have the ability to use their internal state to process sequential inputs (such as sentences), thus keeping a “memory” over previously discovered information. Moreover, Long Short-Time Memory (LSTM) and Gated Recurrent Unit (GRU) are special kinds of RNNs with specialized “gates” that allow for better handling of the information flow, especially with longer sequences. For example, Bagnall [27] trains a RNN-based language model for each author in the PAN task, where the single author’s texts are represented by separate outputs that rely on a shared recurrent state; the classification is thus obtained based on the ability of the single author’s language model to predict the unknown document. Convolutional Neural Networks (CNNs) have also proven to be very effective, both at word level [245] and especially at character level [263], and are extremely fast. Jafariakinabad et al. [141] combine both CNN and LSTM for processing documents that have been converted into sequences of POS tags.

However, NNs are often employed to extract a latent representation of the documents under scrutiny, that can be later used for comparison, akin to the process of the similarity-based methods. To this goal, many works employ siamese NNs: they are made of two identical channels that process one document each, and were initially employed to verify the paternity of signatures [57]. The two outputs can be directly used to solve the SAV task, for example by computing the absolute difference of the two representations [18], the cosine distance [247] or the Euclidean distance [48]. These latent representations can also be fed to a classifier that is an entirely different model. For example, Ding et al. [84] exploit four representations, each one capturing different aspects of the documents (topical, lexical, character-level, syntactic), to solve both SAV via cosine distance, and various AP tasks by training a LR. Recently, the new state of the art in extracting features from texts has been achieved by transformers [291], DL models that use attention mechanisms to capture contextual information and to process sequences in parallel. In particular, BERT

(Bidirectional Encoder Representations from Transformers) models [159] are especially appreciated by the AAn community, both for AP [231] and AId [29, 94].

Instead of representing each document with an embedding, Huertas-Tato et al. [136] extract a transformer-based “authorship embedding” for each author, confronting it with the latent representation of the unknown document.

Hosseinia and Mukherjee [134] propose to employ the reconstruction loss given by an autoencoder that tries to reconstruct a document  $d_2$  from the encoded signal of document  $d_1$  in order to determine whether the two documents share the same author. The various losses given under different feature sets are fed to a Gaussian Naive Bayes or a decision tree for classification.

As already explained, the use of DL methods is particularly difficult in cultural heritage settings, especially when dealing with dead or less widespread languages. However, some fruitful attempts have been made. For example, Tearle et al. [279] create a committee of NNs with the same configuration but random initial weights, and explore its performance with the classic problems of the *Federalist Papers*, and to study the distinction between the English authors Shakespeare and Marlowe. Chandrasekaran and Manimannan [69] apply a Probabilistic NN to a dataset made of three contemporary Tamil scholars that started to write anonymous articles during the oppression of the British Regime. Finally, Vainio et al. [288] investigate the Ciceronian paternity of four disputed documents by training a CNN. Yamshchikov et al. [298] employ a fine-tuned BERT model originally trained for Modern Greek to address resource scarcity in Ancient Greek texts. Their study not only corroborates the hypothesis that the three pseudo-Plutarchian texts examined were not authored by Plutarch, but also identifies three distinct stylistic profiles, two of which appear to originate from the Alexandrian context (2nd–3rd century CE).



## Chapter 3

# The *Epistle to Cangrande*

The *Epistle to Cangrande*, from now on EpistleC, is an important work from the corpus of the Italian poet Dante Alighieri (1265-1321), whose paternity has been vigorously disputed in the last centuries (we offer an overview of its characteristics and of the state of the debate in Section 3.2). We here present our study regarding the authorship of this document so important for our cultural heritage: we subject the document to an AAn investigation, by creating an ad-hoc dataset of medieval Latin texts and by developing a specific AId system, which we thoroughly test and then apply to the specific authorship problem.

In the next paragraphs, after reviewing some related work (Section 3.1) and the recent history of EpistleC (Section 3.2), we present our experimental setting (Section 3.3), describing the dataset we have assembled for this study and the details of the algorithm we employ. We finally show the results of our investigation (Section 3.4) and discuss the outcome, commenting on the major issues faced within this work (Section 3.5).

The research presented in this chapter was published in three articles by Corbara et al. [7, 8, 76], and is based on the preliminary work conducted in the M.Sc. Thesis by Corbara [75].<sup>1</sup>

### 3.1 Related work

As stated in Section 1, documents of unknown or disputed authorship are rather common in our cultural heritage, especially in centuries-old traditions, where the testimony of the true author may easily have been lost or altered. In particular, a number of recent works have tackled problems of AAn for the Latin language.

Kestemont et al. [165] address an AA task characterised by two disputed documents written in medieval Latin and three possible authors – the well-known Christian mystic Hildegard of Bingen, her secretary Guibert of Gembloux, and Bernard of Clairvaux. They employ a PCA-based approach on the frequencies of 65 function words.

In a later work, Kestemont et al. [167] tackle another AA problem concerning parts of the *Corpus Caesarianum*; they include in the candidate set Caesar, his general Aulus Hirtius, and three other unidentified authors. The methodologies they adopt is based on the comparison of an author’s profile (where an author’s profile is defined as the centroid of the vectors corresponding to that author’s known texts) with the document of disputed authorship. They employ two different techniques, namely: the distance between the vectors representing the author’s profile and the disputed document, and a generic implementation of IM [176] (the method is discussed in Section 2.2). They use word unigrams and character

---

<sup>1</sup>The code to replicate the experiments is available at: <https://doi.org/10.5281/zenodo.3903235>.

$n$ -grams as features, and they test their systems beforehand on: i) the datasets from the Authorship Verification track at PAN-2014; and ii) a corpus of historic Latin authors.

An approach that similarly exploits the concept of author profile can be found in the study by Tuccinardi [284] regarding the authenticity of one of Pliny the Younger’s letters. In particular, the author employs the “simplified profile intersection”, a similarity measure that uses the size of the intersection among the profile of the unknown document and the profile of the target author’s production, which is computed by counting the  $n$ -grams in common between them (more on this in Section 2.2). In order to find the model with the best discriminating power between Pliny’s and non-Pliny’s writing, additional fragments of letters from Cicero and Seneca are employed.

IM is again the method of choice in the work by Stover et al. [276]. Here, a newly found Latin document is investigated in a SAV setting, where word unigrams and bigrams are used as features. Ultimately, the only textual pair that receives a satisfying positive score is the one consisting of the disputed document and *De Platone* by Apuleius, hence strongly supporting the hypothesis that Apuleius may be the also the author of the disputed document.

The study by Vainio et al. [288] is the only one, among the ones we consider here, that employs a DL algorithm. In particular, the authors train both a SVM and a CNN for an AV task, consisting of recognising Cicero’s written style against the styles of the background authors, and then use the two trained classifiers to classify four disputed documents. They conduct various experiments with POS-grams and character 5-grams. Their dataset, which is freely downloadable, present the largest number of authors among the works discussed in this section (44 authors), including anonymous and pseudo-authors; this is thanks to the wide timeframe considered, which goes from the 1st century BC to the 5th century AD.

The work by Kabala [156] instead is the only one that focuses on medieval Latin specifically. It performs SAV on two texts, the *Translatio s. Nicolai* and the *Gesta principum polonorum*. In particular, through the studies on four different datasets, the author seeks to understand whether the alleged authors of the two documents, the so-called Monk of Lido and Gallus Anonymous respectively, are actually the same person. The study is conducted by classifying both texts with respect to the author classes within each dataset, using 9 distance metrics and the LR algorithm. Each dataset counts between 39 and 116 texts dating from the 10th to the 12th centuries, written by between 15 and 22 different authors. These are the only datasets of medieval Latin texts that are freely available to the public among the ones we have surveyed in this section.

While the above works focus on cases of uncertain paternity, AAn methodologies might also be applied to documents of certain authorship, e.g., in order to study possible stylistic influences among authors. In the study by Forstall et al. [101], for example, the goal is to verify a supposed influence by Catullo on the poetry of Paul the Deacon. They train a SVM with samples of Catullo’s writings (in a typical AV setting), employing various kinds of  $n$ -grams as features. A document highly influenced by Catullo, thus bearing many similarities to his style, would then receive a high classification score by the AV system.

In Table 3.1 we summarise the works discussed in this section. In general, it should be noted that the creators do not subject their datasets to a thorough cleaning from information extraneous to the authors’ production. In particular, citations of other authors (i.e., pieces of text written by someone other than the author of the citing text) are seldom removed (in some cases, only the most extensive ones are). This may hamper the study, since the cited text “contaminates” the citing text, at least concerning the stylistic analysis [152, p. 248].

Table 3.1: Main characteristics of published works on AAn for the Latin language. The works are in alphabetical order by first researcher; for each paper, we show the task the researchers tackle, the number of authors they consider (**#authors**), the methods and features they employ, the resource they gather their dataset(s) from, and whether the dataset is freely available to the public or not.

Paper	Task	#authors	Methods	Features	Resource	Available
Forstall et al. [101]	AV	7	SVM	Functional n-grams (on text and metric) and low-probability n-grams	Transcriptions and Tesseræe	No
Kabala [156]	SAV	15–22	Distance metrics and LR	250 most frequent words	Patrologia Latina and Latin Library	Yes
Kestemont et al. [165]	AA	3	PCA	65 function words	Brepols Publishers	No
Kestemont et al. [167]	AA	29 (dev) 3 (test)	Distance metrics on author’s profile and IM	Word unigrams and char n-grams	Latin Library	Yes
Stover et al. [276]	SAV	36	IM	Word unigrams and bigrams	Brepols Publishers and Latin Library and Patrologia Latina	Partially
Tuccinardi [284]	AV	3	Simplified Profile Intersection	Character n-grams	[unspecified]	No
Vainio et al. [288]	AV	44	SVM and CNN	POS-tags, word and char n-grams	Latin Library and Bibliotheca Augustana	Yes

### 3.2 The history of the *Epistle*

EpistleC is the thirteenth of the letters from Dante’s epistolary corpus survived until our times. Written in medieval Latin,<sup>2</sup> it is addressed to Can Francesco della Scala, known as Cangrande I, the ruler of the Italian cities of Verona and Vicenza at the beginning of the 14th century. Traditionally, it is divided into 90 paragraphs, where two thematic portions can be defined: the first one (Ep. 1–13, from now on Ep1) is the dedicatory section, with proper epistolary characteristics, while the second one (Ep. 14–90, from now on Ep2) contains an exegesis of the *Divina Commedia*, Dante’s *magnum opus*, including an *accessus* and an *expositio textus* of the first few lines of the third cantica, the *Paradiso*. Among Dante’s letters, it became rather renowned over the centuries, especially because it would be the only analysis we have received from Dante Alighieri about his own masterpiece.

However, more recently the authenticity of EpistleC has been questioned, starting from the study by Scolari [258, pp. 12–21]. Generally speaking, the academic community can be divided among those who consider EpistleC authentic, and those who consider it a spurious work, either in its entirety or at least in its second portion. To support these claims, on the one hand, some scholars, like Baranski [28], Casadei [63], Nardi [210], point out numerous places in the composition where the logical sequence is ambiguous, rusty, or even incoherent within itself or with other writings by Dante. Moreover, Hall and Sowell [117] have noticed that there is a deep dissimilarity between the dedicatory and the exegetical section of the letter, in their themes, style and rhythm. Even figuring out a time-frame when the letter could have been written is not a trivial problem. On the other hand, others, like Azzetta [311, pp. 280–1] Jenaro-MacLennan [144, pp. 67–8], note that there is a lexical coherence spreading through the entire EpistleC, and an inner cohesive logic. Additionally, Ascoli [23] observes that a forger would have followed more closely Dante Alighieri’s prose, and that the dissimilar style should thus paradoxically be seen as a further proof of paternity. Many also note that the author of EpistleC offers some non-traditional and potentially controversial explanations for some exegetical and linguistic issues, which could be the proof of a prominent author, since a common forger would have probably stayed on more ordinary and “safer” grounds. For a more comprehensive outline over this authorship debate, see the analysis by Sasso [254], Azzetta [311].

<sup>2</sup>Medieval Latin is different from classical Latin in a number of ways; e.g., it is more generous in its use of prepositions and conjunctions, and it has a more regular syntax.

The debate is still far from being settled [64], and so, in order to gain a fresh perspective over the unresolved problem, thus offering yet another useful tool for investigation to the scholars, we resolved to apply the methodologies and technologies of AAn to the mystery of EpistleC.

### 3.3 Experimental setting

In the next paragraphs, we present the setting of our study regarding the authorship of EpistleC: we describe the dataset we have collected, divided among Ep1 and Ep2 (Section 3.3.1), and we introduce *MedieValla*,<sup>3</sup> the algorithm we employ for authorship experiments on medieval Latin texts (Section 3.3.2). We finally describe our experimental protocol (Section 3.3.3).

#### 3.3.1 MEDLATIN: Two datasets for medieval Latin

As explained in Section 3.2, scholars are torn regarding the paternity of the two distinct portions of EpistleC, Ep1 and Ep2. Therefore, we tackle the AId problem as two separate AId sub-problems, one for each portion. Given the different nature of the two portions, we built two separate datasets: we will refer to them as MEDLATINEPI (where EPI refers to the epistolary nature of the texts contained therein, in line with Ep1) and MEDLATINLIT (where LIT stands for literary, in line with the characteristics of Ep2), respectively. We refer to the two datasets together with the name MEDLATIN. Their composition and characteristics are detailed in Appendix B.1.

The datasets bring together (in preprocessed form) a set of texts that were not readily available, since some of them were not available in digital form, while others lay scattered across different electronic formats and different digital libraries. This is the reason why we make them available to the research community. Note that, even though we here employ MEDLATIN as dataset for our study regarding EpistleC, it can certainly be used as dataset for medieval Latin AAn researches that does not necessarily involve EpistleC, or as benchmarks for general-purpose, language-agnostic AAn systems. Hence, by making MEDLATIN available to the public, we hope to support the general AAn community, especially the researchers who suffer from the shortage of available and already-processed documents.

#### 3.3.2 *MedieValla*, a tool for Latin AV

As stated, we aim to study the paternity of the two portions of EpistleC with the methodologies of AId; in particular, we focus on the alleged Dantean authorship. As such, we define our problem as a binary AV task, where the possible classes are Dante and NotDante. We tackle this AV problem via a methodology we call *MedieValla*, whose implementation details we present here.

First, we further pre-process the textual documents. We remove the explicit citations, either in Latin or other languages, and we segment each resulting text into shorter texts, so as to increase the overall number of labelled examples, while reducing their average size. This is necessary because ML processes require a significant number of training examples, regardless of their length. In particular, for each text:

- we identify the sentences that make up the text (via the NLTK library, see Appendix D); if a sentence is shorter than 8 words, we merge it with the next sentence (or the previous sentence, if it is the last sentence of the text);
- we create sequences of 3 consecutive non-overlapping sentences, and consider each of these sequences as a labelled text, and assign it the author label of the text from which it was extracted. We call

---

<sup>3</sup>The name *MedieValla* is a combination of “medieval” and the last name of Lorenzo Valla (1407–1457), one of the first authorship researchers recorded in history.

these newly-made texts “segments”.

We use both the original texts in their entirety and the segments that we create as labelled examples. Thus, the number of labelled examples increases from 294 to 1,310 for MEDLATINEPI and from 30 to 12,772 for MEDLATINLIT.

We also lower-case the entire text and remove the punctuation marks. The reason why we ignore punctuation marks is that they were not inserted by the authors (punctuation was absent or hardly coherent in ancient manuscripts), and are thus an expression of the editors, not the original authors.

We then convert each labelled text into a vector of features. The set of features we use is divided into six feature types (for a discussion regarding these features, and more generally regarding feature design, see Section 2.1):

1. **Character  $n$ -grams**: we consider each character  $n$ -gram (for  $n \in \{3, 4, 5\}$ ) that occurs in the training set as a potential feature;
2. **Word  $n$ -grams** we consider each word uni-gram and bi-gram ( $n \in \{1, 2\}$ ) that occurs in the training set as a potential feature;
3. **Function words**: the relative frequency of each function word (from a list of 74 Latin function words);<sup>4</sup>
4. **Verbal endings**: the relative frequency of each verbal ending (from a list of 245 regular Latin verbal endings);<sup>4</sup>
5. **Word lengths**: the relative frequency of each word length (from 1 to 23 characters);
6. **Sentence lengths**: the relative frequency of each sentence length (from 3 to 70 words).

The vector space results from the union of all of these features. In order to deal with the high dimensionality of the feature space, we subject the features resulting in a sparse distribution (character  $n$ -grams and word  $n$ -grams) to a process of dimensionality reduction. First, we perform feature selection via the Chi-square function (see Appendix A.2); in our experiments, we select the best 10% character  $n$ -grams and the best 10% word  $n$ -grams. We then perform feature weighting via the Tfidf function (see Appendix A.3). For MEDLATINEPI the number of resulting features is 16,101, while for MEDLATINLIT the resulting features are 86,924.

The six subsets of features described above have very different cardinality: the numbers of features contained in types (1) and (2) is very high (in both cases it ranges in the tens of thousands features), while the numbers of features contained in types (3), (4), (5), (6) are fixed (there are 74, 245, 23, 68, features in each of these groups, respectively), and are much smaller than the two previous ones. This means that the latter groups may end up being overwhelmed by the former groups, in terms of their contribution to the classification process. In order to avoid this, we individually normalise each of the six feature subsets via L2-normalisation, so that each of the six vector subspaces they define have unit norm.<sup>5</sup>

As learning algorithm, we use LR, as implemented in the `scikit-learn` package (see Appendix D). We train each binary classifier by optimising the hyperparameter  $C$  (the inverse of the regularisation

---

<sup>4</sup> The list is available in the code repository.

<sup>5</sup> This means that the contribution of, say, a character  $n$ -gram, ends up being smaller than the contribution of, say, a word length, because there are more character  $n$ -grams than word lengths. This does not prevent the classifier from uncovering which among the features are the most important (these might well include some character  $n$ -grams) or least important (these might well include some word lengths) though, since the classifier attempts to find the linear combination of feature weights that best classifies the documents.

strength) via stratified 10-fold cross-validation (10-CV), using a grid search on the range of values in the log-space  $\{10^i\}_{i=-3}^3$ . We use a variant of stratified 10-CV called “grouped” stratified 10-CV, that prevents different segments from the same document (“group”) to end up in different folds; by doing this, the classifier never unduly benefits from testing on segments of a document whose segments have been seen during training.

We also briefly report on some additional experiments for which we use other learning algorithms: SVM (for which we have optimised the hyperparameter  $C$  in the same manner as for LR) and Multinomial Naïve Bayes (MNB) (for which we have optimised the hyperparameter  $\alpha$  for the range of values in the log-space  $\{10^i\}_{i=-7}^0$ ).

### 3.3.3 Experimental protocol

Before applying *MedieValla* to the problem of *EpistleC*, we subject the system to a “leave-one-out” (LOO) validation test, which consists of predicting, for each dataset  $D \in \{\text{MEDLATINEPI}, \text{MEDLATINLIT}\}$ , for each author  $A^*$  in the set of authors  $\mathcal{A}$  represented in  $D$ , and for each document  $d_i \in D$ , whether  $A^*$  is the author of  $d_i$  or not, where the prediction is issued by an “ $A^*$  vs.  $\bar{A}^*$ ” binary classifier trained on all labelled texts (i.e., segments and entire documents) from  $D \setminus \{d_i\}$ . This means that all labelled texts from documents in  $D \setminus \{d_i\}$  originating from author  $A^*$  are used as positive training examples while all labelled texts from documents in  $D \setminus \{d_i\}$  originating from authors other than  $A^*$  are used as negative training examples, thus as  $\bar{A}^*$ . Note that:

- In order to faithfully reproduce the operating conditions of an AV system, we use only entire documents as test examples, not segments.
- In order to avoid any overlap between training examples and test examples, when document  $d_i$  is used as a test document, we exclude from the training set all the segments derived from  $d_i$ .
- In order to avoid any overlap between the training phase and the test phase, both the feature selection step and the parameter optimisation step are performed not on the entire dataset  $D$ , but on  $D \setminus \{d_i\}$ . This means that the entire cycle <feature selection + parameter optimisation + classifier training> is repeated for each document  $d_i \in D$ , for both *MEDLATINEPI* and *MEDLATINLIT*.
- We do not train classifiers for authors for which we have only one text in  $D$ , since this would entail experiments in which the author is not present both in the training and in the test set;<sup>6</sup> as a result, the texts of these authors are used only as negative examples in experiments focused on other authors. Ultimately, this means that we train binary classifiers for 5 authors of *MEDLATINEPI* (all authors except those from the collection of Petrus de Boateriis, since this collection is a miscellanea of authors) and 6 authors for *MEDLATINLIT*; this leads to  $5 \times 294 = 1470$  predictions for *MEDLATINEPI* and  $6 \times 30 = 180$  predictions for *MEDLATINLIT*, where each prediction is the result of a different cycle consisting of <feature selection + parameter optimisation + classifier training>.<sup>7</sup>

---

<sup>6</sup>For this same reason, we bypass the hyperparameter optimisation phase in cases in which we only have 2 positive documents and one of them is acting as the held-out document. This would cause the training set to have only one positive document (plus fragments), and this would eventually force one of the trainings (as generated via 10-CV) to be devoid of any positive example (since in the “grouped” variant of stratified 10-CV the fragments of a document are always within the same fold as the full document). In those (few) cases, we resort to a LR that is moderately regularised (we set  $C = 0.1$ ) in order to avoid overfitting the one and only positive document; likewise, for SVM we also set  $C = 0.1$  and for MLB we set  $\alpha = 0.001$ .

<sup>7</sup>Since we use 10-CV for hyperparameter optimisation and explore a grid of 7 hyperparameter values, our experimentation consists of roughly 115,000 trainings per learner (we report experiments for 3 learners).

Table 3.2: Summary results of our AV experiments on the MEDLATINEPI and MEDLATINLIT datasets employing different learning algorithms. We show the  $F_1^M$ ,  $F_1^\mu$  and Acc values.

	MEDLATINEPI			MEDLATINLIT		
	$F_1^M$	$F_1^\mu$	Acc	$F_1^M$	$F_1^\mu$	Acc
LR	<b>.954</b>	<b>.969</b>	<b>.989</b>	<b>.572</b>	<b>.615</b>	<b>.944</b>
SVM	.944	<b>.969</b>	<b>.989</b>	.383	.435	.928
MNB	.760	.933	.976	.310	.357	.900

Table 3.3: Results of the validation test on MedieValla. We show the  $F_1$  value and the Acc value obtained by the authorship verifier trained via LR for the specified author.

	Author	$F_1$	Acc
MEDLATINEPI	Clara Assisiensis	1.000	1.000
	Dante Alighieri	.857	.990
	Giovanni Boccaccio	.980	.997
	Guido Faba	.946	.973
	Pietro della Vigna	.986	.986
MEDLATINLIT	Benvenuto da Imola	.800	.967
	Boncompagno da Signa	.333	.867
	Dante Alighieri	.500	.933
	Giovanni Boccaccio	.800	.967
	Giovanni del Virgilio	.000	.933
	Nicola Trevet	1.000	1.000

### 3.4 Results

As explained, we first validate MedieValla on a series of AV tasks on the authors of the two datasets. We employ  $F_1$  and Acc as metrics (see Appendix A.1).

In Table 3.2 we briefly compare the performance of our learner of choice (LR) with the performance of two other popular learning algorithms, specifically SVM and MNB. LR clearly is the best algorithms in this regards, outperforming the other learners in both MEDLATINEPI and MEDLATINLIT.

In Table 3.3 we show the results obtained by MedieValla on the different authors of the datasets. Note that, as evident from the  $F_1$  and Acc columns, there is a lot of variability in the scores (especially for  $F_1$ ) across different authors for the same dataset. There are at least three possible explanations for this:

- For some authors there are more (positive) training data than for other authors. Since AV consists of a different binary classification task for each author, this means that it will be easier (other things being equal) to conduct the AV task for the former authors than for the latter.
- Some large differences in  $F_1$  values are due to the idiosyncrasies of the  $F_1$  measure. For instance, the authorship verifier for Giovanni del Virgilio, when asked to verify the 30 texts in MEDLATINLIT, returns 2 false negatives and 28 true negatives. Despite having correctly predicted 28 out of 30 times (the “vanilla accuracy” result is  $\text{Acc}=28/30=0.933$ ), the verifier obtains a  $F_1$  value of 0 because there are no true positives (see Equation 2), i.e., none of the two texts actually by Giovanni del Virgilio were correctly predicted as by him.
- Even with an equal amount of training data for each author, we might observe varying accuracy

Table 3.4: Results of the application of the two authorship verifiers “Dante vs. NotDante” to the two portions of the *Epistle to Cangrande*. Columns 4 and 5 recall (from Tables 1 and 2) the  $F_1$  and Acc values that the “Dante vs. NotDante” verifiers have obtained in the experiments of Section 3.4.

	Binary decision	Posterior probability	$F_1$	Acc
Ep1	No	.367	.857	.990
Ep2	No	.022	.500	.933

results because some authors’ style may be inherently more challenging to identify.<sup>8</sup>

Interestingly, an analysis of the individual ⟨author, document⟩ classification decisions<sup>9</sup> shows that there are no systematic mistakes, but just a few, scattered individual ones. In particular, it never happens that there are two or more incorrectly classified documents with the same true author  $A_x$  and with the same predicted author  $A_y$ , with  $A_x \neq A_y$ ; in other words, there are no systematic mistakes that would indicate an extreme similarity in style between two authors  $A_x$  and  $A_y$ . One of the reasons for this is that the mistakes made by our verifiers are very few, i.e., only 26 out of 1650 AV decisions (16 out of 1470 for the MEDLATINEPI experiments and 10 out of 180 for the MEDLATINLIT experiments) are incorrect.

At the end of this evaluation stage, we can affirm that MedieValla is a reliable AV system for the authors involved in this study, and we can then proceed to apply the system to the problem of EpistleC, more precisely to its two distinct portions Ep1 and Ep2 (Section 3.4.1).

### 3.4.1 Is Dante the real author of the *Epistle to Cangrande*?

As introduced in Section 3.2, with this study we attempt to solve the puzzle of the *Epistle to Cangrande*, which means verifying if the letter addressed to Cangrande della Scala was indeed written by Dante Alighieri.

After running the validation experiments described in the previous paragraphs, for each of the two datasets we have retrained MedieValla for the author Dante Alighieri (i.e., so that the positive examples indicate authorship by Dante and the negative examples indicate authorship by someone other than Dante), rerunning the entire cycle ⟨feature selection + parameter optimisation + classifier training⟩ on the corresponding dataset; we have then applied the classifier derived from MEDLATINEPI to the first portion of the epistle (Ep1) and the classifier derived from MEDLATINLIT to the second portion (Ep2).

The results of the application of the two classifiers to the problem of EpistleC are reported in Table 3.4. These results show that out MedieValla believes that both portions of the *Epistle to Cangrande* are the work of a forger.

Once applied to Ep1, the “Dante vs. NotDante” verifier trained on MEDLATINEPI returns a posterior probability of 0.367: this means that the verifier believes that Ep1 is not written by Dante Alighieri (since this probability is  $< 0.500$ ), and it is moderately confident about this fact, since its “degree of confidence” is  $(1 - 0.367) = 0.633$ . Moreover, as can be seen in Table 3.3, the Dantean verifier has also proved very accurate ( $F_1 = 0.857$ , Acc = 0.990) once tested on MEDLATINEPI via LOO. These two facts, altogether, make a fairly convincing case for the non-Dantean authorship of Ep1.

Concerning Ep2, the “Dante vs. NotDante” verifier trained on MEDLATINLIT returns a posterior probability of 0.022: this means that the verifier believes that Ep2 is also not by Dante (since this probability is  $< 0.500$ ), and is extremely confident about this fact, since its degree of confidence is  $1 - 0.022 = 0.978$ . As can be seen in Table 3.3, this verifier has proved reasonably accurate ( $F_1 = 0.500$ ,

<sup>8</sup>We investigate this intuition in another study, see Section 4.2.5.

<sup>9</sup>We provide, in spreadsheet form, the list of all ⟨author, document⟩ classification decisions as taken by MedieValla, as well as the results displayed previously, at the following link: <https://doi.org/10.5281/zenodo.4298503>.

Table 3.5: Result of the application of the authorship verifier “Dante vs. NotDante” to the *Epistle to Henry VII*. Column 4 recalls (from Table 3.3) the  $F_1$  value that the “Dante vs. NotDante” verifier has obtained in the experiments of Section 3.4.

	Binary decision	Posterior probability	$F_1$	Acc
EpistleH	No	.026	.857	.990

Acc = 0.933) once tested on MEDLATINLIT via LOO. These two facts support the hypothesis that Ep2 was not written by Dante as well.<sup>10</sup>

### 3.4.2 Addendum: The *Epistle to Henry VII*

Pellegrini [225], in a recent study regarding an epistle addressed to emperor Henry VII and signed by Cangrande della Scala (*Epistle to Henry VII*, from now EpistleH), conjectures that the real author of the epistle could be Dante Alighieri himself. In order to investigate this hypothesis, we apply the same “Dante vs. NotDante” authorship verifier that we have applied to Ep1 (Section 3.4.1) (i.e., MedieValla trained with the documents of MEDLATINEPI, which are of epistolary nature) to EpistleH. The results of this application are shown in Table 3.5.

MedieValla strongly rejects the hypothesis that EpistleH may have been written by Dante; in fact, it believes that EpistleH is by someone other than Dante with probability  $(1 - 0.026) = 0.974$ . Together with the fact that this verifier has shown very high accuracy ( $F_1 = 0.857$ , Acc = 0.990) in the experiments of Section 3.4, this result does not support the theory that EpistleH is the work of Dante.

## 3.5 Discussion

In this chapter, we have described MEDLATINEPI and MEDLATINLIT, two new datasets of cultural heritage texts written in medieval Latin by 13th- and 14th-century (mostly Italian) literates and labelled by author, that we make publicly available to researchers working on the AAn field. We hope these datasets will become valuable tools for researchers investigating techniques for AAn, and especially for texts written in Latin or medieval Latin.

We also make available the source code of MedieValla, an AV tool that we have built in order to work on two important case studies, i.e., the real paternity of the *Epistle to Cangrande*, allegedly written by Dante Alighieri but believed by some to be a forgery, and of the *Epistle to Henry VII*, whose Dantean paternity has been recently theorized. We also describe in detail the preliminary experiments in which we have applied MedieValla to MEDLATINEPI and MEDLATINLIT. We hope that the availability of these two datasets and of our AV tool will support researchers interested in AV in their investigations.

Despite the strong theoretical grounds this project builds upon, it presents some important limitations. In particular, we identify four major weak points, which we break down into four problems (already hinted in Section 1.3):

- **Lacking features engineering.** In this project, we employ features that are nowadays considered standard in the literature, such as function words, and character and word  $n$ -grams. However, especially these latter ones, are known to exploit topical information, which is highly undesirable

<sup>10</sup>Note that a classifier that obtains  $F_1 = 0.500$  is *not* equivalent to a classifier that returns random decisions; in fact, a completely clueless classifier for which half of the positives are true positives while the other half are false negatives, and half of the negatives are true negatives while the other half are false positives, on MEDLATINLIT would obtain a value of  $F_1 = (2TP)/(2TP + FP + FN) = (2 \cdot 1)/(2 \cdot 1 + 14 + 1) = 0.058$ . The  $F_1 = 0.500$  result for the “Dante vs. NotDante” authorship verifier is the result of generating 1 TP, 1 FP, 1 FN, and 27 TNs, i.e., 28 correct predictions out of 30 total predictions, on dataset MEDLATINLIT.

in AAn studies, for reasons already discussed in Section 2.1. This brings about the question: is it possible to devise a new feature type, that is effective in capturing the authorial style of written documents, while at the same time being topic-agnostic? We refer to this as **Problem 1**. In this regards, we propose a new feature type, based on the rhythm of the discourse. We experiment with its application for the Latin and Spanish languages, exploiting the concepts of “syllabic quantity” and “syllabic stress”, together with psycholinguistics-based features, in Chapter 4.

- **Limited dataset size.** As described in Section 3.3.2, we segmented the documents of the dataset in order to obtain a substantial amount of text examples, even though shorter in length, in order to enable the employment of ML algorithms. Even by doing this, the resulting amount was not vast; for this same reason, we avoided the use of DL algorithms, since we assumed they would have performed suboptimally (indeed, see also the results in Section 4.1.4). In Chapter 5, we approach this issue, which we refer to as **Problem 2**, through a different route: we try to make a better use of the information contained in the (limited) dataset, by exploiting a vectorial representation of the textual documents that does not depict a single document, but pairs of documents, quadratically augmenting the amount of examples at our disposal.
- **A supposed forgery.** As discussed in Section 3.2, the scholars opposing the authorship by Dante Alighieri postulate that the *Epistle to Cangrande*, and particularly Ep2, is actually the work of a forger, i.e., somebody who wanted to make their work look as if written by Dante. Within this hypothesis, we can very well expect the supposed forger to try to imitate Dante’s writing style, in order to make the falsification more believable. However, in this research we do not take into account this possibility. Admittedly, this is also due to the fact that we do not have any testimony of a known Dantean forger at our disposal, but we treat each document as if it was a real manifestation of each author’s style, the *Epistle to Cangrande* being no exception. However, more in general, it is rare in AId studies that a researcher can strike off with full certainty the possibility of an adversary, especially in cultural heritage settings. We refer to this as **Problem 3**, and we try to tackle it in Chapter 6.
- **No explanations for the predictions.** Aside from a very basic visualization method for the classification scores (that we display in Section 7.1), in this project we did not invest in offering an explanation for the classification prediction of MedieValla, despite the fact that this is arguably the most important aspect for the domain specialist. Without it, the scholar is unable to make much use of the output of a AAn system, since they cannot enquire on the reasons why the classification system returned a certain decision. We refer to this as **Problem 4**, and in Chapter 7 we offer an in-depth analysis of this issue, by evaluating some existing methodologies for the specific case of the cultural heritage setting.

## Chapter 4

# New topic-agnostic features for Authorship Analysis

Even though many different feature types have been proposed over the decades (see Section 2.1), it has been frequently noted that certain AAn methods run a high risk of capturing the domain of the text under study, rather than its style. In the terminology of statistics: domain-dependent features act as confounding variables. This means that, as pointed out for example in Bischoff et al. [45], Halvani et al. [121], if topic-dependent features are used, an authorship system (even a seemingly good one) might not really perform AAn as desired, but might unintentionally perform topic identification, unwittingly leveraging not the stylistic peculiarities of an author but the linguistic peculiarities typical of a certain topic. Of course, it is true that some authors confine their written production to very restricted domains, but it would clearly be a poor decision to classify a document as written by author  $A^*$  only because  $A^*$  often or always writes about the same topic as the document under consideration. Word  $n$ -grams and character  $n$ -grams particularly suffer from this problem [268]; in fact, the good performance they usually deliver in AId tasks may be due to the fact that the datasets employed are far from being topic-neutral. It would hence be good practice to avoid, as much as possible, the use of “topic markers” when implementing AAn algorithms.

With this goal in mind, various techniques can be employed. One possibility consists of using only topic-agnostic features, such as function words or syntactic features [119, 141]. A second possibility consists of actively masking topical content via a so-called “text distortion” approach [270, 289].

In this chapter, we tackle the problem that we define as **Problem 1** in this Thesis; that is, the problem of defining an effective feature type for AAn that does not carry topical information, and we propose the use of rhythmic features to achieve this goal. In particular, in Section 4.1 we define our proposal for a topic-agnostic rhythmic feature type for the Latin language based on the concept of *syllabic quantity* (from now on, SQ) [277], and in Section 4.2 we test the adaptation of this idea for the Spanish language exploiting the concept of *syllabic stress* (from now on, SS); in the latter, we also experiment with other topic-agnostic features based on psycholinguistic traits. Finally, in Section 4.3 we draw some conclusion regarding the outcome of this investigation.

### 4.1 Rhythmic features for Latin

In this section, we experiment with the idea of using SQ in order to derive an additional set of stylistic features for AId, and in particular for the task of AA for Latin prose documents. In the Latin language, syllables can be *long* or *short* based on their *quantity*, and peculiar sequences of long and short syllables

were used by Latin authors as metric (i.e., rhythmic) patterns. Our idea to use these sequences as features for performing AId is based on accumulated evidence suggesting that some Latin authors show a preference, more or less conscious, for specific rhythmic patterns obtained by specific sequences of long and short syllables, even in prose texts (see later Section 4.1.2). In order to assess the plausibility of this idea, we run a number of experiments, using three different dataset, in which we evaluate the impact of SQ-based features on the accuracy of the AA classifiers obtained with two different learning algorithms.

In the next paragraphs, we first present some related work (Section 4.1.1), then in Section 4.1.2 we detail our proposed methodology regarding the use of SQ as rhythmic features, including some theoretical background. In Section 4.1.3 we describe our experiments, including the datasets we employ and the experimental protocol we follow, while in Section 4.1.4 we show the results of the experiments.

The research presented in this section is published in an article by Corbara et al. [6].<sup>1</sup>

### 4.1.1 Related work

In the AAn field, the idea of employing prosodic features is not a new one. Of course, their most natural use is in projects focused on poetry, such as in the study of Neidorf et al. [212] on the Old English verse tradition, or in the already cited investigation by Forstall et al. [101] on the supposed influence of Catullus on Paul the Deacon’s writings. Nevertheless, rhythmic or prosodic features have also been employed in AAn of prose text. However, these works usually consist of the study of word repetitions, like the anaphora (the repetition of a word, or a sequence of words, from a previous sentence at the beginning of a new sentence) [178], or are based on mapping the texts into the corresponding sequences of sounds before extracting the  $n$ -grams, such as in the research by Forstall and Scheirer [100], where the authors employ the CMU Pronouncing Dictionary for the conversion.<sup>2</sup>

Syllables have been used as base units in other AId works [264] and more generally in other NLP tasks, such as poem generation [308]. These works, while close in nature to our project, explore a linguistic dimension different from the one we aim to capture with the study of SQ.

Some studies closer to ours employ the distribution of accent in order to derive rhythmic features for AA in English; see Section 4.2.1 for a more complete survey of the research on this subject. Accentuation (as explained in Section 4.1.2), is related to the concept of SQ (at least in the Latin language); however, to the best of our knowledge, SQ has never been employed for any AId tasks concerning prose texts.

### 4.1.2 A brief introduction to syllabic quantity, and how to extract it

The Latin language is based on *syllables*, i.e., sound units a single word can be divided into, which can be thought of as oscillations of sound in the pronunciation of the word. Every Latin word has as many syllables as it has vowels or diphthongs.<sup>3</sup> Generally speaking, a Latin word is divided into syllables according to the following rules:

- A single consonant and the vowel that follows it belong to the same syllable, e.g., “pater” (“father”) is divided into two syllables as “pa-ter”;
- Two adjacent consonants belong to two adjacent syllables, e.g., “mitto” (“I send”) is divided as “mit-to”, and “arma” (“weapons”) is divided as “ar-ma”;

---

<sup>1</sup>The code to replicate the experiments is available at: [https://github.com/silvia-cor/SyllabicQuantity\\_Latin](https://github.com/silvia-cor/SyllabicQuantity_Latin).

<sup>2</sup>Available at: <http://www.speech.cs.cmu.edu/cgi-bin/cmudict>. E.g., with this conversion the word “reason” becomes “R IY1 Z AH0 N”.

<sup>3</sup>Diphthongs are combinations of two vowels that count as a single, long vowel. In Latin, only the combinations “ae”, “au”, “ei”, “eu”, “oe”, “ui” are diphthongs.

- Compounds generate different syllables, e.g., “abest” (“he/she/it is missing/away”), being composed of the preposition “ab” (“from”) and the verb “est” (“he/she/it is”), is divided as “ab-est”.

A syllable is characterized by its *quantity*, which refers to the amount of time required to pronounce it. Specifically, a syllable can be *long* or *short*, and this is determined first and foremost by the quantity of its vowel, and then by the consonant sounds that follow it. In fact, a single vowel has its own quantity, which in turn depends on the structure of the word, or on its etymology; so, for example, a vowel before another vowel (when the two do not form a diphthong) is short, while a vowel originating from a contraction, such as “nil”, contracted from “nihil” (“nothing”), is long. In the study of SQ, long vowels are traditionally marked with a *macron* (ā), while short vowels are (only sometimes) marked with a *breve* (ă). A syllable is said to be short if it contains a short vowel; it is said to be long “by nature” if it contains a long vowel (or a diphthong), and it is said to be long “by position” if it contains a short vowel followed either by two consonants or by a double consonant (“x” or “z”).

Note that the explanations offered here regarding both the syllabification and the quantification rules for Latin are rather generic, and many more detailed rules exist, with exceptions and specific cases, making the study of prosody all but a trivial matter; see the expositions by Ayer [25], Ceccarelli [67], Harrison [125], Sturtevant [277], for a more complete discussion on these topics.

It is well known that classical Latin (and Greek) poetry followed metric patterns based on SQ, i.e., on well-chosen sequences of short and long syllables. In particular, syllables were combined in what is called a *foot*, and a series of feet composed the *metre* of a verse. For example, one of the most renowned meters is the *dactylic hexameter* (employed, among others, by Virgil in his *Aeneid*), which consists of 6 *dactyl feet* (each consisting of a long and two short syllables), with the possibility of a substitution with a *spondee* (consisting of two long syllables) in most positions; additionally, the sixth foot can be a *trochee* (consisting of a long and a short syllable). An example of a dactylic hexameter is:<sup>4</sup>

$$- \cup \cup | - \cup \cup | - - | - - | - \cup \cup | - X ||$$

Arma vi|rumque ca|nō, Trō|iae quī| p̄rīmus ab| ōrīs||

Similar metric schemes were followed also in many prose compositions, in order to give a certain cadence to the discourse, and to focus the attention on specific parts of it. In particular, the end of sentences and periods, known as *clausula*, was deemed to be especially important in this sense. Orators such as Cicero were particularly aware of the effects of such rhythmic endings, as in the example below (consisting of a *molossus*, i.e., a foot consisting of 3 long syllables, followed by a *cretic*, i.e., a foot consisting of the sequence “long-short-long”):

$$- - - | - \cup - ||$$

cōnsulum scelus, cupiditās, egestās, au|dācia!||

During the Middle Ages, Latin prosody underwent a gradual but profound change, that also propagated to romance languages: the concept of SQ lost relevance in favour of the *accent*, or *stress*. As a matter of clarification, both phenomena, SQ and accent, were present in both classic and medieval Latin. However, stress did not have a role in rhythmic composition in classic Latin (as we have seen), and was pronounced with a higher *pitch*. Instead, medieval Latin speakers gradually stopped “hearing” the quantities of word syllables, in favour of a higher *intensity* given by stress, even though a stressed syllable typically requires also a longer time to be pronounced. The modern consequences of this process can be seen, for example,

<sup>4</sup>Regarding the following notation, “-” stands for a long syllable, “∪” stands for a short syllable, and “X” stands for an *anceps*, which can be either a short or a long syllable; the vertical bar indicates where one foot ends and the other begins, and the double vertical bar indicates where the dactylic hexameter ends. Concerning this example, we observe that the particle “-que” is always a short syllable, and that an “i” between vowels has a consonant function.

in Italian poetry, where a verse is characterized by the number of syllables (but not their quantities) and the positions of the accents. Moreover, Latin accentuation rules are largely dependent on SQ; so, for example, words longer than two syllables are accented on the next-to-last syllable if this syllable is long, e.g., “amícus” (“friend”), otherwise they are accented on the third-to-last syllable, e.g., “dómīnus” (“master”).

In the middle of these transformations, medieval writers retained the classical importance of the *clausula*, although it now followed the change in paradigm and became based on stress rather than quantity. Stress-based rhythmic patterns are known as *cursus*. We can distinguish the following three main types of *cursus*:<sup>5</sup>

<i>Cursus planus</i> :	- +   + - +	illum dedúxit
<i>Cursus tardus</i> :	- +   + - + +	íre tentáverit
<i>Cursus velox</i> :	- + +   + + - +	saécula saeculórum

Many scholars (see e.g., Janson [143], Keeline and Kirby [158]) have shown that certain authors preferred specific types of rhythmic patterns, and that differences can be detected even between authors who cannot be assumed to consciously care for such rhythmic patterns at all, both in metric based on quantities [158, p. 187] and in metric based on accents [143, p. 20]. This is an important point: many authors consistently show a certain (more or less conscious) preference for specific rhythmic constructions, even if they do not follow the prosodic canons of the time.

An author’s use of certain rhythmic patterns might thus play an important role in the identification of that author’s style. In fact, it has already been used in non-computational studies of debated authorship, such as the case of EpistleC [117] and the treatise *Quaestio de Aqua et Terra* [283], both traditionally attributed to Dante Alighieri.

Given these premises, the goal of this study is to investigate whether features based on SQ can be profitably employed for computational AA in Latin prose texts. This seems reasonable also for medieval Latin, since accents are heavily based on SQ, as already explained. Note also that features derived from SQ are also content-agnostic (e.g., a sequence of syllables such as “long-short-long” can stand for hundreds or thousands of different 3-syllable sequences), and thus they could be a valuable tool for AAn problems, since they avoid the risk of unwanted influence from the topic.

An important part of the computational system that needs to be assembled in order to carry out our experiments on SQ is a module that extracts SQ from a given piece of Latin written text, in order to generate SQ-based features that can be used for classification. Since developing such a module would be a major endeavour, due to the complexity of Latin prosody already mentioned here, we decided to use an off-the-shelf tool, chosen among those that are publicly available.

We considered the tool that resulted from the *Cursus in Clausula* project [267]: this tool is a web application that not only extracts all the forms of *cursus* from an uploaded text, but it also allows performing some statistical analysis on it. However, this tool analyses only the final portions of periods and sentences (following the definitions of *clausula* and *cursus*). We believe instead that it would be more profitable to extend our analysis to the entirety of the document, and not only to the final portions of periods and sentences, since this would highlight potential rhythmic preferences of an author in creating the more general structure of the discourse; additionally, we should remember that there are rhythmic rules also for the beginning of the sentence [143].

We eventually implemented our SQ extraction module by using the `Classical Language Toolkit` library (see Appendix D): among many tools for the study of ancient languages, it offers specific ones

---

<sup>5</sup>Regarding the following notation, “-” stands for a stressed syllable and “+” stands for an unstressed syllable. See for example Janson [143], Oberhelman and Hall [218] for an in-depth analysis of this stylistic technique.

Table 4.1: Example results of the two modules from the CLTK library on a Latin prose text.

<b>Original text</b>	Quo usque tandem abutere Catilina patientia nostra. Quam diu etiam furor iste tuus nos eludet.
<b>Macronizer</b>	quō usque tandem abūtēre catilīna patientia nostra . quam diū etiam furor iste tuus nōs ēlūdet .
<b>Scanner</b>	[-U-U--UUU-UUU-UU-X, -UUU-UU-UU-----X]

for the study of Latin prosody, in particular the two modules **Macronizer** (which places a macron  $\bar$  over long vowels, e.g.,  $\bar{o}$ ) and **Scanner** (which produces a sequence of the traditional symbols denoting the quantity of a syllable, i.e., short, long, and end of sentence). The output of each functionality is shown in the respective entry in Table 4.1.

### 4.1.3 Experimental setting

As already stated, in this study we focus specifically on the AA problem. Therefore, given document  $d$  we aim to assign it to exactly one class among a set  $\mathcal{A} = \{A_1, \dots, A_m\}$  of candidate classes, i.e., possible authors. We evaluate the contribution of SQ-derived features when added to other topic-agnostic features, using two different learning algorithms and three datasets; thus, the quality of SQ-derived features is inferred from the difference between the performance of a method without SQ-based features and the performance of the same method with SQ-based features.

In this section, we outline the procedural details of our experiments. In particular, we describe the datasets we employ and how we preprocess them (Section 4.1.3.1), the topic-agnostic features we use (Section 4.1.3.2), and finally the different learning algorithms and the evaluation protocol (Section 4.1.3.3).

#### 4.1.3.1 Datasets

Currently, there is no Latin dataset that is considered a standard in AA studies, and even the available ones are few.

In this research, we perform our experiments on three Latin datasets:

- **LATINITASANTQUA**. This dataset was assembled by us, in order to best suit our requirements; see Appendix B.2 for full information regarding this dataset.
- **KABALACORPUSA**. This dataset was created and used by Kabala [156]; see Appendix B.3 for full information regarding this dataset.
- **MEDLATIN**. This dataset was developed for the case study of the *Epistle of Cangrande*, already treated in Chapter 3; see Appendix B.1 for full information regarding this dataset. For this project, we combine the two sub-datasets (MEDLATINEPI and MEDLATINLIT) together. Moreover, from MEDLATINEPI we exclude the texts from the collection of Petrus de Boateriis, since the collection consists of a miscellanea of different authors, often represented by just one epistle each; as such, this collection is hardly useful for our goals. We delete the quotations from other authors and the parts in languages other than Latin, both marked in the texts.

We automatically pre-process all the documents in the three datasets in order to clean them, as much as possible, from spurious information and noise. In particular, we delete headings, editors' notes, and other meta-information, if present. We delete symbols (such as asterisks or parentheses) and Arabic numbers, since they are likely bibliographical information inserted by the editor. We normalise punctuation marks:

Table 4.2: Information regarding the datasets we use. For each dataset we report the number of items (# entire texts, or # segments) and the mean number of words for each item (mean length), both for the entire texts and for the resulting segments.

LatinitasAntiqua	entire texts	# entire texts	90
		mean length	40,170
	segments	# segments	23,219
		mean length	156
KabalaCorpusA	entire texts	# entire texts	39
		mean length	34,389
	segments	# segments	7,882
		mean length	170
MedLatin	entire texts	# entire texts	294
		mean length	3,985
	segments	# segments	6,028
		mean length	194

we delete commas, and we replace all question marks, exclamation marks, semicolons, colons and suspension points with full periods. We do this because punctuation was absent or hardly coherent in ancient manuscripts, hence the punctuation we see in current editions follows modern habits, and is mostly due to the editor, not to the author [282, p. 57]. However, we retain full periods in order to be able to divide the text into sentences. We lowercase the text, and then we normalise it: we exchange (i) all occurrences of character “v” with character “u”; (ii) all occurrences of character “j” with character “i”; and (iii) every stressed vowel with the corresponding non-stressed vowel.<sup>6</sup>

As a final step, we divide each text into sentences, where a sentence is made of at least 5 distinct words (we attach shorter sentences to the next sentence in the sequence, or to the previous one in case the sentence is the last one in the document). Each non-overlapping sequence of 10 consecutive sentences is what we call a “segment”. These segments are the textual samples that we give as input to our learning algorithms and classifiers; the final amount of segments for each of our three datasets is displayed in Table 4.2, along with some additional information regarding the composition of each dataset.

#### 4.1.3.2 Topic-agnostic features: Base features and distorted views

In order to obtain a topic-agnostic representation of the text, we follow both routes sketched at the start of this chapter: using only topic-agnostic features, or actively masking topical content via a so-called “text distortion” approach.

- **Base features.** We employ a set of features that are well-known and widely used in the AA literature, and generally acknowledged as being topic-independent (for a discussion regarding these features, and more generally regarding feature design, see Section 2.1). In this research they will act as a common base for each classifier, with other types of features being added to them (according to a process based on the learning method used, see Section 4.1.3.3). We call this set BaseFeatures (from now on: BFs); it is composed of the following feature types:
  - **Function words:** the relative frequency of each function word; the entire list can be found in Appendix C;
  - **Word lengths:** the relative frequency of each word length, from a minimum of 1 up to a maximum of 25 characters;
  - **Sentence lengths:** the relative frequency of each sentence length, from a minimum of 1 up to a maximum of 100 individual words;

<sup>6</sup>The reason for these actions are the same as discussed in Footnote 6 in Appendix B.1.

- **POS-tags:** the relative frequency of each POS tag. We extract POS tags using the `TnT tagger` module of the `CLTK` library (see Appendix D); the extraction results in 12 POS tags being assigned to our data.

For each feature type we compute a matrix  $f \times t$ , where  $f$  is the number of segments in the set and  $t$  is the number of features of the specific type, and we further scale each vector to unit norm (see Footnote 5 in Section 3.3.2). Given the four resulting matrices, we concatenate them in a single final matrix  $f \times 217$ , where  $(80 + 25 + 100 + 12) = 217$  is the total number of BFs.

- **Distorted views.** In addition to the SQ-based encoding (see Section 4.1.2), we experiment with the four text “distortion” (i.e., masking) methods presented by Stamatatos [270], which aim to preserve the stylistic characteristics of the document while at the same time hiding its topical content.<sup>7</sup> Each such method generates what Stamatatos [270] calls a *distorted view* (from now on: `DistView`). Given a list  $F$  of function words,<sup>8</sup> the four `DistViews` are:

- **Distorted View - Single Asterisk (`DistViewSA`):** every word not included in  $F$  is masked by replacing it with an asterisk (\*).
- **Distorted View - Multiple Asterisks (`DistViewMA`):** every word not included in  $F$  is masked by replacing each of its characters with an asterisk (\*).
- **Distorted View - Exterior Characters (`DistViewEX`):** every word not included in  $F$  is masked by replacing with an asterisk (\*) each of its characters except the first and the last one. Note that one- and two-character words thus remain unaffected. As stated by Stamatatos [270], `DistViewEX` is based on many psychological studies underlining that “exterior” (i.e., first and last) characters are more important than “interior” characters for reading comprehension; thus, the positions of these characters appear to have a higher importance in how we process words (both in the reading activity and, one might imagine, in the writing activity). Additionally, the end of a word and the start of the following word might create sound effects that certain authors may want to avoid (e.g., using a word that begins with the same character as the end of the preceding word), or, conversely, to actively employ in their writing.
- **Distorted View - Last 2 (`DistViewL2`):** every word not included in  $F$  is masked by replacing with an asterisk (\*) each of its characters except the last two. Note that one- and two-character words thus remain unaffected. Underlying `DistViewL2` is the attempt to capture morpho-syntactic information (e.g., number, tense) that is often encoded in language via word suffixes.

The logic behind these masking methods is to remove any type of topic-dependent information from the representation of the text, while at the same time retaining topic-independent information. Some of the information that is retained with these methods is independent from word order, such as function words, word lengths, sentence lengths, starting characters and ending characters of words, and their frequencies; some of this information is already captured by the BFs. However, some of the information that is retained is instead positional (i.e., dependent on word order). Examples are:

- for `DistViewMA`, `DistViewEX`, `DistViewL2`: the lengths of words that follow (or precede) specific function words, the lengths of words that are used as the first (or last) word of the

---

<sup>7</sup>Note the distinction between the text encoding procedures and the distortion methods by Stamatatos [270]: while the former “translate” the text in order to focus on a certain topic-agnostic linguistic property, the latter expunge the information regarded as topic-related, including portions of words.

<sup>8</sup>We employ the same list of function words reported in Appendix C.

- sentence, the lengths of words that follow (or precede) short words (or long words), and their frequencies;
- for DistViewEX: the frequencies with which a word begins (or ends) with certain characters, the frequencies with which a word that ends with a certain character is followed by a word that begins with another given character, etc.;
- for DistViewL2: the frequencies with which a word that ends with a given sequence of two characters is followed by a short word, etc.

Namely, these DistViews allow capturing phenomena that transcend the lexical level, thus pertaining to the structure of the sentence. This is especially important when using learning mechanisms that employ the entire sequence as input (such as NNs, see Section 4.1.3.3), and that are thus sensitive to positional information.

We thus obtain five different “encodings” of each document, i.e., the one representing SQ (see Section 4.1.2) and the four DistViews described by Stamatos [270]. From these five encodings we can extract various kinds of features, an operation for which we also need to take into account the specific learning algorithm we adopt; this is unlike the set of BFs, since BFs form a matrix which remains the same for both learning algorithms used in this project. We discuss the feature extraction methods for SQ and DistViews in the sections specific to each learning method in the following Section 4.1.3.3. Note that we also use the combination of the features extracted from all four DistViews; we call such a combination ALLDistViews.

### 4.1.3.3 Experimental protocol

We assess the performance of the different classifiers on a given dataset by randomly splitting the dataset into a training set (containing 90% of the data) and a test set (10% of the data). After performing this split, we further remove from the training set 10% of its data, in order to use it as validation set.<sup>9</sup> This tri-partition of the dataset into training set / validation set / test set is stratified, meaning that the class distribution in the original dataset is preserved in all three resulting subsets. For a given dataset, the tri-partition is the same for all the systems being tested.

As evaluation measure, we use the well-known  $F_1$  function, in its  $F_1^\mu$  and  $F_1^M$  variants (see Appendix A.1). As anticipated, we aim to compute the difference in performance between a method employing SQ-based features and the same method without SQ-based features, using this difference as indicator of the contribution of SQ to the AA task for Latin prose texts. To this aim, we also compute the statistical significance of the aforementioned difference, via McNemar’s paired non-parametric statistical hypothesis test [200]. Since the test applies to binary results (instead of categorical results), we convert the predictions of the two methods of interest into binary values, where 1 stands for a correct prediction and 0 stands for a wrong prediction. We take 0.05 as the confidence value for statistical significance.

---

<sup>9</sup>Note that, since our texts are split into segments, it might well be that segments belonging to the same text end up both in the training data and in the test data. It might be argued that, as a consequence, the attribution task is unduly facilitated, since patterns encountered in the test data may have already been encountered in the training data. However, even assuming this to be the case, the task would be equally facilitated for a system that employs SQ-based features and for a system that does not employ them, which means that the comparison between them is hardly going to be invalidated. We also want to stress that enforcing a stricter separation between training data and test data would be hardly possible for us, since for several authors that we consider (e.g., 15 authors out of 25 in the *LatinitasAntiqua* dataset) we only have 1 text (typically, an entire book that gives rise to several thousand segments). Such problems arise routinely when dealing with ancient texts, as already mentioned in Section 1.2.3, since in these cases the number of available labelled texts may be extremely limited, and is anyway upper-bounded by the known production of the authors considered. The above-mentioned lack of a stricter separation between training and test data can thus be found in many AAn works that deal with cultural heritage texts [52, 103, 165, 167, 196, 284].

Table 4.3: Number of features extracted in the different feature sets (**Feat. set**) from the combination of training and validation sets for each dataset. For the SQ-grams, we report the total number of features extracted before feature selection.

Feat. set	LatinitasAntiqua	KabalaCorpusA	MedLatin
BFs	217	217	217
SQ	3,242	2,929	2,592
DistViewMA	474	456	469
DistViewSA	470	453	466
DistViewEX	1,245	1,028	1,014
DistViewL2	1,416	1,139	1,109

For this experiment we consider two learning algorithms, SVM and NN; we explain in detail the setup of both in the two sections that follow. As we show, in order to feed the five different encodings of the text (the SQ-based encoding and the four DistView-based maskings) to the SVM algorithm, we extract character  $n$ -grams from the encoded texts. On the other hand, our NN architecture does not necessarily require this step, since it is possible to pass the encoded texts to the NN as they are.

- **Support Vector Machines.** The SVM implementation we employ in this study is the `LinearSVC` module from the `scikit-learn` package (see Appendix D). This implementation employs by default a linear kernel and a one-vs-rest multi-class strategy, which has been found by the developers similar in performance to the method by Crammer and Singer [78] (a standard algorithm for turning binary SVM into multiclass SVM), but less demanding in terms of computational cost.

We experiment with various SVM-based classifiers, each characterized by a specific feature set. In order to feed the five different encodings of the text (the SQ-based encoding and the four DistView-based maskings) to a SVM, we extract character  $n$ -grams from the encoded texts. Given the matrix of BFs, any additional feature set is simply concatenated to it, so that the  $f \times 217$  matrix of BFs becomes an  $f \times k$  matrix, where  $(k - 217)$  is the number of additional features, i.e., character  $n$ -grams extracted from the various encodings of the text. We show the number of features extracted as BFs and from each encoding in Table 4.3.

In particular, for the SQ encoding we use character  $n$ -grams (where a “character” is one of the three SQ symbols “U”, “-”, “X”) with  $n \in [\alpha \dots \beta]$ . We set  $\alpha = 3$  since many metric feet in Latin poetry are based on 3 syllables, and we set  $\beta = 7$  because the most important cursus rhythms are based on schemes between 5 and 7 syllables long (see Section 4.1.2). On the other hand, for each of the DistView distortions we follow Stamatatos [270] and use character 3-grams that appear at least 5 times in the training set.

For all the features derived from the five encodings, we perform feature weighting via `Tfidf` (see Appendix A.3). Since the SQ encoding gives rise to a large number of features (from now on, SQ-grams),<sup>10</sup> we perform filter-style feature selection (i.e., we retain the  $k$  top-scoring features) using Chi-square (see Appendix A.2) as the feature scoring function. We do not perform feature selection on the BFs feature set and on the  $n$ -grams extracted from the DistViews maskings.

We perform the optimisation of two parameters: the SVM parameter  $C$ , which sets the trade-off between the training error and the margin, and the feature selection factor  $r$ , which is the fraction of SQ-grams that are retained as a result of the feature selection phase. In particular, our approach is as follows:

1. We create a list of possible configurations for the classifier, where a configuration is made of a

<sup>10</sup>For instance, for the `LatinitasAntiqua` dataset the SQ encoding gives rise to a number of features one order of magnitude larger than either the `DistViewSA` or the `DistViewMA` masking.

possible value for parameter  $C$  (we explore the range of values in the log-space  $\{10^i\}_{i=-3}^{i=3}$ ) and a possible value for parameter  $r$  (we explore the range of values  $\{0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 1.0\}$ , where 0.0 denotes a method that does not employ SQ-grams). Thus, a possible configuration is ( $C = 10, r = 0.5$ ).

2. For each configuration we train a classifier on the training set, and assess its performance on the validation set.
  3. Using the configuration that has scored the highest value of  $F_1^M$  in validation, we train the final classifier from the union of training set and validation set.
  4. We assess the final classifier on the test set, computing the values of  $F_1^M$  and  $F_1^\mu$ .
- **Neural Networks.** The NN architecture for DL that we implement for this work is based on the idea of using each of the 6 feature sets we explore (1 based on the SQ-derived encoding, 4 based on the DistView-derived maskings, and 1 based on BFs) as a separate input source for the network; this approach is inspired by the idea of the multi-channel architecture developed by Ruder et al. [245]. Specifically, the network consists of parallel input “branches”, each one processing a single feature set, with the outputs of the parallel branches being combined in a single decision layer. Note that the number of branches thus depends on the feature sets one wants to combine: in case of the matrix of BFs and a single encoding (for example, only the SQ-based encoding), the network is made of only two branches (one for BFs and one for the SQ-based encoding), and so on.

The following setup for each input branch is the result of a series of preliminary tests that we have run on the validation sets using different combinations of layers. We have found that, for this task, a simple sequence of CNNs are more effective than a combination of CNN and LSTM layers, or a combination of CNN layers and an attention mechanism [291]. This might be due to the limited size of the datasets, which, when the number of parameters increases, does not allow the network to properly generalise (we recall that NNs usually require very large amounts of training data). Character-level CNNs have been proven to work efficiently for text classification in general [305], and on AA tasks in particular [263].

Each of the five encodings of a document are passed to the corresponding branch, and are independently padded so that every string in the same input batch has the same length. A branch is made of a total of 5 layers. Given  $m$  classes (i.e., authors) and an input batch of  $b$  segments (we set the batch size to 64), the layers are as follows:

- **Embedding layer.** It performs character embedding on the distorted text (the embedding size is set to 32).
- **CNN block.** There are 2 CNN layers, applying a convolutional operation with 128 kernels of size 3 and 5 respectively. After each layer, we apply a ReLU activation, and perform shrinkage on the output with a max-pooling operation. We concatenate the outputs of the two layers, and finally perform a dropout operation (with the drop probability set to 0.5) on the result.
- **Dense layer.** It applies a ReLU operation over the input. This final layer is made of  $m$  neurons.

A slightly different branch is devoted to the BFs, which are passed as input in the form of a matrix. For this branch, the feature matrix is processed by two dense layers with a ReLU activation, with a dropout process in-between.

Therefore, each branch returns a  $b \times m$  matrix of probabilities, where  $b$  is the number of segments in the input batch and  $m$  is the number of classes. These outputs are then stacked together, so as

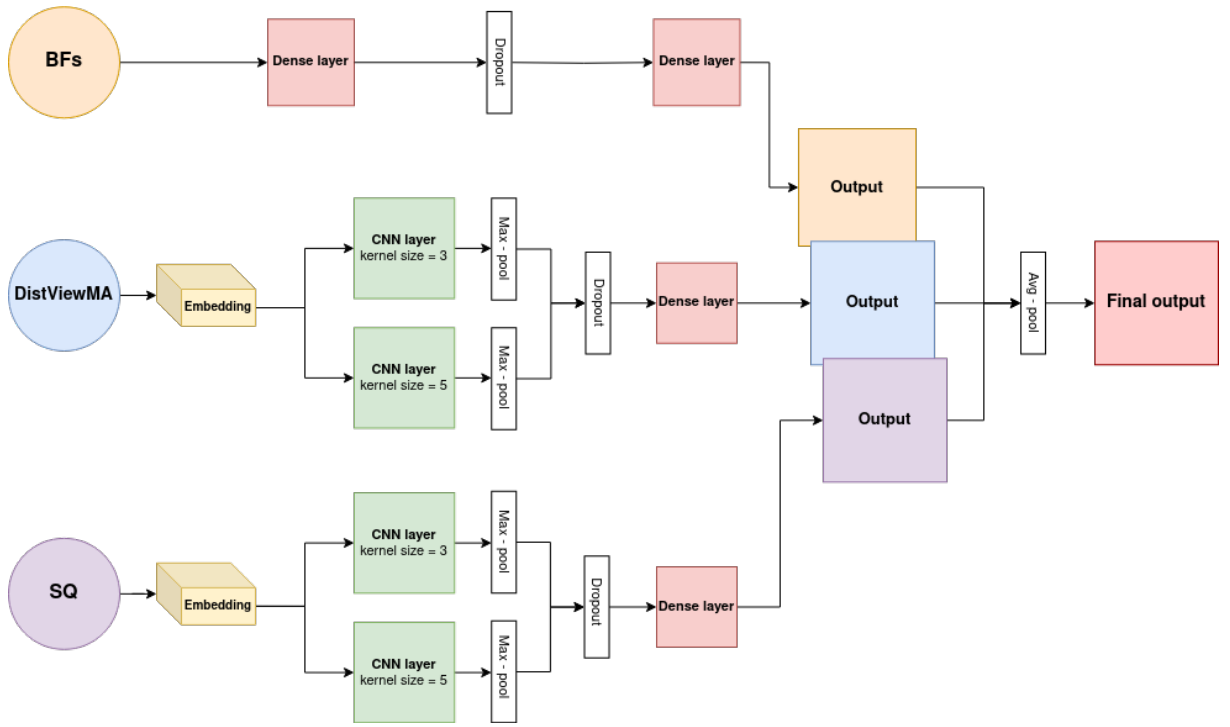


Figure 4.1: Scheme of the NN architecture developed. In the example displayed, it is instantiated with the BFs + DistViewMA + SQ setup.

to obtain a decision matrix of dimension  $b \times m \times c$ , where  $c$  is the number of branches employed. An average-pooling operation is applied on the dimension  $c$ , so that, for each class, the average value of the decisions of the different branches is finally obtained, yielding a  $b \times m$  matrix. The final decisions, i.e., the author classes predicted for the input segments, are obtained via a final dense layer which applies a softmax (for training) or argmax (for testing) operation over the class probabilities.

Menta and Garcia-Serrano [202] employ a similar approach in a SAV setting: in their approach, two branches of different size process the information coming from TfIdf-weighted character  $n$ -grams and punctuation marks respectively, after which the two outputs are combined into a single series of layers that finally yields the classification result. Instead, in our method each branch computes a separate classification decision, and the decisions are then averaged. This is somewhat akin to exploiting a committee (or ensemble) of classifiers, an approach which has proven more effective than single classifiers in various settings [11, 235]. In the AA literature, Tearle et al. [279] have employed a similar committee of NNs, each one with a different feature set; however, instead of hard-counting the vote of each NN, as they do, we choose a “softer” approach and average the probabilities computed by each branch, akin to what is done by Muttenthaler et al. [209]. A scheme of our NN architecture is displayed in Figure 4.1.

We perform the training of the NN with the traditional backpropagation method; for this, we employ Cross Entropy as the loss function, and the Adam optimizer [171]. In particular, our approach is as follows:

1. We train the NN on the training set.
2. We assess the performance of the NN on the validation set in terms of  $F_1^M$ .
3. We repeat the previous steps for 100 epochs without improvement before triggering an early stop (the maximum number of training epochs is 5,000, but, in the experiments we perform,

early stopping always occurs after some hundreds epochs).

4. We train the model that has scored the highest  $F_1^M$  in validation (before early stopping) with 10 final epochs on the combination of training and validation sets.
5. We assess the final result of the NN on the test set in terms of  $F_1^M$  and  $F_1^\mu$ .

A legitimate concern that may arise when comparing NN models is the possibility that the superiority of some methods (as quantified via any evaluation measure) is caused by the mere presence of a higher number of parameters, and not to the supposedly more informative content of the additional branches. In order to check if this is the case, we have developed a dummy channel named FAKE, acting as control, which allows a fair comparison in terms of number of parameters. This channel is different not in its architecture, but in the encoded text it processes: in fact, the text is encoded with a “fake” version of the SQ encoding. Specifically, each word is transformed as follows: given a number of syllables  $s$  (which we approximate to the number of characters divided by 3), each syllable is replaced by a random SQ symbol (“U”, “-”, or “X”). We collect all these transformations in a dictionary, so that a certain word is always replaced by the same sequence of symbols, thus making the transformation uninformative, but not equal to random noise. The NN cannot learn anything useful from this kind of information, hence the result obtained with this channel acts as baseline against which to compare the effect of the SQ channel. Note that we test the FAKE channel only in conjunction with all the DistView channels (i.e., in the ALLDistViews setup), purely as a control experiment.

#### 4.1.4 Results

As already mentioned, we perform two batches of experiments, each with a different learning algorithm (SVM and NN). In each batch, we compare various models with SQ-based features against the same models without SQ-based features, in order to assess the performance gain (if any) obtained by the addition of these features. The results of the two batches of experiments are displayed in Table 4.4.

Regarding the experiments with SVMs, in most cases the accuracy of the classifier is improved by the addition of the SQ-grams. This effect is very substantial in *LatinitasAntiqua*, where the presence of the SQ-grams always improves the accuracy, irrespectively of feature set and evaluation measure, and in half the cases does so in a statistically significant sense. The improvement remains considerable in *KabalaCorpusA*, although the SQ-grams cause a (statistically insignificant) decrease in performance in one case. Finally, it is difficult to give a proper assessment for the *MedLatin* dataset, since the SQ-grams result in a statistically significant difference in only two cases, one with a negative outcome and one with a positive outcome. We conjecture that this might be due to rhythmic patterns suffering from the limited size of the documents and from the authors being located in a small timeframe, thus exhibiting similar prosodic habits; this is true for the *KabalaCorpusA* dataset and even more so for the *MedLatin* dataset.

In general, it is worth noting that:

- for 4 out of 6 combinations (evaluation measure, dataset), the best-performing feature set involves SQ-based features;
- for 8 combinations (feature set, dataset), SQ-based features bring about a statistically significant improvement in performance, while a statistically significant deterioration in performance due to the introduction of these features is observed only in one case.

Regarding the experiments with NNs, it is clear that our network is not able to efficiently employ SQ-based features, which mostly do not bring any difference in performance: the difference is statistically

Table 4.4: Results of our experiments with the SVM and NN learning algorithms. Pairs of experiments, one without SQ-based features and the other with SQ-based features, are reported on two consecutive rows. For each experiment we report: the values  $F_1^M$  and  $F_1^\mu$ , derived on a single train-validation-test split, the percentage improvement ( $\Delta\%$ ) resulting from the addition of SQ-based features, and the results of the McNemar statistical significance test (**M**) against the baseline. The best result obtained for the given dataset and evaluation measure for each learning algorithm is in **bold**.

		LatinitasAntiqua					KabalaCorpusA					MedLatin					
		$F_1^M$	$\Delta\%$	$F_1^\mu$	$\Delta\%$	<b>M</b>	$F_1^M$	$\Delta\%$	$F_1^\mu$	$\Delta\%$	<b>M</b>	$F_1^M$	$\Delta\%$	$F_1^\mu$	$\Delta\%$	<b>M</b>	
SVM	BFs	.620		.721			.659		.692			.764		.814			
	BFs + SQ	.718	+15.79	.801	+11.05	*	.690	+4.74	.717	+3.66	*	.721	-5.65	.814	—		
	BFs + DistViewMA	.718		.803			.678		.716			.754		.819			
	BFs + DistViewMA + SQ	.751	+ 4.70	.823	+ 2.47	*	.715	+5.44	.752	+4.96	*	.691	-8.33	.791	-3.44		
	BFs + DistViewSA	.693		.790			.675		.712			.760		.824			
	BFs + DistViewSA + SQ	.750	+ 8.36	.826	+ 4.52	*	.709	+5.05	.743	+4.27	*	.780	+2.60	.842	+2.21	*	
	BFs + DistViewEX	.842		.891			.821		.828			<b>.904</b>		.915			
	BFs + DistViewEX + SQ	.856	+ 1.67	.898	+ 0.68		.825	+0.54	.834	+0.77		.795	-12.02	.894	-2.36	*	
	BFs + DistViewL2	.849		.894			.860		.868			.817		.902			
	BFs + DistViewL2 + SQ	.857	+ 0.86	.901	+ 0.77		.852	-0.99	.866	-0.29		.835	+2.17	.915	+1.47		
	BFs + ALLDistViews	.879		.922			.855		.864			.846		<b>.947</b>			
	BFs + ALLDistViews + SQ	<b>.888</b>	+ 0.99	<b>.927</b>	+ 0.56		<b>.887</b>	+3.74	<b>.885</b>	+2.35	*	.834	-1.42	.935	-1.23		
	NN	BFs	.648		.787			.609		.688			.709		.803		
		BFs + SQ	.701	+ 8.18	.806	+ 2.41	*	.618	+1.45	.689	+0.18		.652	-8.10	.796	-0.83	
BFs + DistViewMA		.723		.840			.697		.731			.747		.844			
BFs + DistViewMA + SQ		.736	+ 1.82	.844	+ 0.46		.677	-2.81	.730	-0.17		.683	-8.65	.829	-1.77		
BFs + DistViewSA		.712		.824			.674		.724			.749		.844			
BFs + DistViewSA + SQ		.763	+ 7.28	.853	+ 3.55	*	.704	+4.40	.729	+0.70		.719	-4.05	.851	+0.79		
BFs + DistViewEX		.817		.885			.794		.825			.798		.887			
BFs + DistViewEX + SQ		.826	+ 1.05	.896	+ 1.27	*	.756	-4.75	.795	-3.69	*	.786	-1.47	.887	—		
BFs + DistViewL2		.798		.874			.766		.797			.789		.886			
BFs + DistViewL2 + SQ		.814	+ 2.11	.890	+ 1.87	*	.779	+1.60	.809	+1.43		.802	+1.61	.889	+0.37		
BFs + ALLDistViews		<b>.854</b>		<b>.913</b>			<b>.838</b>		<b>.857</b>			<b>.818</b>		<b>.909</b>			
BFs + ALLDistViews + FAKE		.839	- 1.71	.907	- 0.75		.819	-2.28	.826	-3.55	*	.800	-2.17	.900	-0.91		
BFs + ALLDistViews + SQ		.836	- 2.08	.909	- 0.47		.819	-2.27	.844	-1.48		.810	-1.02	.904	-0.55		

significant only for 6 combinations (channel, dataset), and the best-performing model is always the one equipped with the all the DistView channels, but without the SQ channel. Its effect is rather similar to the addition of the FAKE channel, which results in a significant deterioration in performance only once.

We hypothesize that this might be due to the NN not having enough data to train with: indeed, the results obtained by the NNs are consistently worse than the results obtained by the SVMs for almost every combination (feature set/channel, dataset); in general, this is not surprising, since deep-learning algorithms are notoriously data-hungry. The only exceptions to this trend can be seen strictly within the LatinitasAntiqua experiments (and the BFs + DistViewMA experiment with the KabalaCorpusA dataset), which is the largest dataset among the ones we employ; contextually, the LatinitasAntiqua is the only dataset where the SQ channel does indeed bring a statistical significant benefit in most cases. Therefore, the other two datasets might not provide enough training information to exploit the power of SQ-based features in a NN environment. We thus deem the SQ-based feature as potentially useful also for NN methods, provided that enough training data is available.<sup>11</sup>

It is also worth noting that the increased accuracy obtained through the use of SQ-based features tends to be lower in methods employing the DistViewEX and DistViewL2 masking methods; this is expected, since these methods reach very high  $F_1$  values, which means that for them the margin of improvement is narrower.

Overall, given these results, we can conclude that the idea of deriving rhythmic features from syllabic quantity, and to apply them to AIId tasks for Latin prose texts, is a fruitful one that is worth to further explore. In particular, it would be important to test this type of features in other AIId (and more

<sup>11</sup>In Section 7.4.2, we show that even a novel transformer model such as RoBERTa indeed appear to capture SQ-related information in its latent representation, and employing it for classification.

generally AAn) tasks besides AA. Moreover, we conjecture that rhythmic features might be useful in AAn problems in languages other than Latin as well; the research in this sense should start from languages linguistically close to Latin, such as Italian or Spanish. In particular, it would be fascinating to study whether modern-day prose authors unconsciously opt for specific rhythmic patterns, to the point of being uniquely recognisable thanks to them. Both this points are the focus of the investigation in the next Section 4.2.

## 4.2 Rhythmic and psycholinguistic features for Spanish

In the work detailed here, we continue the research started in Section 4.1 and delve deeper on the subject.

First, given the promising results of SQ-based features for the Latin language, we deem worthwhile to extend the investigation of topic-agnostic rhythmic features for AAn on other languages as well, in order to see the extent to which similar considerations can be applied to languages outside of Latin. Since Spanish is a Romance language, and as such derives from Latin, we opt for this language as a starting point for this research. In this case we encode the rhythmic patterns of prose texts by exploiting the concept of SS, which gained relevance over the concept of SQ in Romance languages, as mentioned in Section 4.1.2. In fact, in the Spanish language each word can be divided into syllables, one of which is *stressed* while the other are *unstressed*, following specific rules of accentuation, like SQ; the stressed syllable is pronounced with higher intensity and intonation [89].

Second, we evaluate the use of rhythmic feature types not only for AA tasks (as in Section 4.1), but also for AV and various AP tasks (by gender, by age, and by political affiliation). Moreover, we compare the results of the SVM classifier in the different tasks with the performance of a BERT-based model fine-tuned on the original texts (and thus potentially learning from topical information),<sup>12</sup> in order to better assess the efficiency of topic-agnostic features in AAn tasks.

Lastly, we experiment with other topic-agnostic features that are based on psycholinguistic traits.

To this aim, we employ a dataset of political speeches from the Spanish Parliament: despite the fact that these textual documents do not belong to the cultural heritage domain in the conventional sense, they provide a rich source of well-annotated textual material. Moreover, we believe that their nature as speeches makes them ideal for analyzing rhythm-based features: although they are typically prepared well in advance (similar to common documents in the cultural heritage domain), they are intended for public recitation, which might prompt authors to pay special attention to the rhythmic patterns of the discourse.

In the next paragraphs, after presenting some related work (Section 4.2.1), we describe our methodology regarding the extraction of rhythmic and psycholinguistic features for Spanish (Section 4.2.2), and then explain our experimental setting (Section 4.2.3) and show the results of our research (Section 4.2.4). Finally, given the strong variance in the results of the AV task, in Section 4.2.5 we take an initial step toward interpreting the impact of different authorial characteristics on this task.

The research presented in this section is published in two articles by Corbara et al. [1, 2].<sup>13</sup>

### 4.2.1 Related work

In Section 4.1.1, we survey the research regarding the use of prosodic information for AAn, and mention that some projects do employ the concept of stress within the English language for prose texts. The work by Dumalus and Fernandez [87] is a pioneering one in this sense: using the CMU Pronouncing

---

<sup>12</sup>Note that, given the mediocre results of the NN in Section 4.1, we do not employ DL algorithms in this project, apart from the BERT-based transformer used for comparison.

<sup>13</sup>The code to replicate the experiments is available at: [https://github.com/silvia-cor/Topic-agnostic\\_ParlaMintES](https://github.com/silvia-cor/Topic-agnostic_ParlaMintES).

Dictionary, they extract the pronunciation of each word and transform it into a “stress string”, where the symbols  $\{0, 1, 2\}$  represent the absence of stress, a primary stress, and a secondary stress in the syllable, respectively. Ivanov et al. [140] improve on this work: since many English words are homographs (i.e., they have the same spelling but different pronunciation and meaning), they select the correct pronunciation, and hence the correct stress string, by studying the POS of the words in the text. Similarly, Plecháč [230] employs the frequencies of “rhythmic types” (where a rhythmic type is a bit string representing the distribution of stressed and unstressed syllables in a line) as features in the study of the attribution problem of *Henry VIII*.

Regarding the study of psycholinguistic features applied to the AAn field, the Linguistic Inquiry and Word Count (LIWC) [226] software is one of the most famous tools for the study of psychological aspects of textual documents. LIWC is built around a word dictionary where each entry is associated with one or more categories related to grammar, emotions, or other cognitive processes and psychological concepts; usually, researchers employ the relative frequency of each LIWC category for text analysis. In particular, it has been profitably used for the characterization of a “psychological profile” or a “mental profile mapping” for AAn studies [52, 105], and also for the analysis of speeches regarding the Spanish political debate [98]. In a similar vein, García-Díaz et al. [104] designed UMUTextStats, a LIWC-inspired tool, and studied its application to AA and various AP tasks (gender, age range, and political spectrum) on a dataset of Spanish political tweets.

## 4.2.2 Syllabic stress and psycholinguistics for authorship studies

As stated, this projects regards the evaluation of features that do not employ topical information for AAn tasks. Concretely, following the methodology in Section 4.1, we propose to generate new encoded versions of the original text by extracting (1) the SS of the text, and (2) the psycholinguistic categories of the words. In order to do so, we proceed in the following manner:

1. **STRESS**: similarly to what we do for SQ (see Section 4.1.2), we extract the SS from the textual documents with a off-the-shelf tool. In particular, we employ the `Rantanplan` library,<sup>14</sup> that directly converts the document into a sequence of stressed and unstressed syllables.
2. **LIWC**: in order to encode the psycholinguistic dimension of the document, we employ the LIWC dictionary.<sup>15</sup> We define three macro-categories from a subset of the LIWC category tags, representing (a) grammatical information, (b) cognitive processes or actions, and (c) feelings and emotions.<sup>16</sup> For each macro-category, we perform a separate text encoding by replacing each word with the corresponding LIWC category tag. Formally, LIWC can be seen as a map  $m : w \rightarrow C$ , where  $w$  is a word token and  $C \subset \mathcal{C}$  is a subset of the psycholinguistic categories  $\mathcal{C}$ . Given a macro-category  $M \subset \mathcal{C}$ , we replace each word  $w$  in a document by the categories  $m(w) \cap M$ . If  $|m(w) \cap M| > 1$ , then a new token is created which consists of a concatenation of the category names (following a consistent ordering). If  $m(w) \cap M = \emptyset$ , then  $w$  is replaced with the “w” symbol. Note that some entries in LIWC have the suffix truncated and replaced with an asterisk (\*), e.g., *president\**; the asterisk is treated as a wildcard in the mapping function, and in case more than one match is possible, the match with the longest common prefix is returned.

<sup>14</sup>Available at: <https://github.com/linhd-postdata/rantanplan>.

<sup>15</sup>We employ the Spanish version of the dictionary, which is based on LIWC2007.

<sup>16</sup>We use the following categories: (a) YO, NOSOTRO, TUUTD, ELELLA, VOSUTDS, ELLOS, PASADO, PRESENT, FUTURO, SUBJUNTIV, NEGACIO, CUANTIF, NUMEROS, VERBYO, VERBTU, VERBNOS, VERBVOS, VERBOSEL, VERBELLOS, FORMAL, INFORMAL; (b) MECCOG, INSIGHT, CAUSA, DISCREP, ASENTIR, TENTAT, CERTEZA, INHIB, INCL, EXCL, PERCEPT, VER, OIR, SENTIR, NOFLUEN, RELLENO, INGERIR, RELATIV, MOVIM; (c) MALDEC, APECT, EMOPOS, EMONEG, ANSIEDAD, ENFADO, TRISTE, PLACER. We avoid employing categories that would repeat information already captured by the POS tags, or topic-related categories (e.g., DINERO, FAMILIA).

Table 4.5: Example of the encodings employed in this project. Note that there is no one-to-one correspondence between syllables and stresses due to linguistic phenomena across word boundaries (e.g., synalepha), which `Rantanplan` accounts for.

<b>Original text</b>	Gracias	.	No	hay	que	restituir	lo	que	no	ha	existido	.
<b>STRESS</b>	+	-	-	-	-	-	-	-	+	-	-	-
<b>LIWC_a</b>	w		NEGACIO	PRESENT	w	w	ELLELLA	w	NEGACIO	PRESENTVERBOS	EL	w
<b>LIWC_b</b>	w		w	w	MECCOG	w	w	MECCOG	w	w		w
<b>LIWC_c</b>	AFFECTEMOPOS		w	w	w	w	w	w	w	w		w

We show an example of these encodings in Table 4.5.

### 4.2.3 Experimental setting

As stated, our focus in this research is to evaluate the employment of rhythm- and psycholinguistics-based features for AId and AP tasks. To this aim, we explore various combinations of feature sets, including other topic-agnostic feature types commonly used in literature. In order to assess the effect of our proposed feature types on the performance, we carry out ablation experiments (in which we remove one feature set from the model) and addition experiments (in which we add one single feature set to the model), and compare our models with the performance of a BERT-based models that has access to topical information.

In this section we outline the procedural details of our experiments. In particular, we describe the dataset we employ (Section 4.2.3.1), the topic-agnostic features we use (Section 4.2.3.2), and finally our experimental protocol (Section 4.2.3.3).

#### 4.2.3.1 Dataset

In this project, we employ the Spanish ParlaMint dataset (see Appendix B.4). Because of their declamatory nature, between the written text and the discourse, these speeches seem particularly suited for an investigation on rhythmic and psycholinguistic traits. Various AAn studies have already been conducted on political documents, especially regarding AP tasks [80]. Apart from lowercasing the text, we did not apply any further pre-processing steps.

In order to have a balanced dataset, we select the parties with more than 300 speeches in the dataset and assign them to the Left, Right, Centre, or Regionalist<sup>17</sup> wing. In particular, we assign PSOE and UP to the Left, PP and PP-Foro to the Right, EAJ-PNV and JxCat-Junts to the Regionalist wing, and only the Ciudadanos (Cs) party to the Centre. We then delete all the speeches that have less than 50 words, and for each wing we select the 5 authors with the most speeches in the dataset.

By doing so, the minimum number of samples per author is 70 (Bal Francés), while the maximum is 467 (Sánchez Pérez-Castejón). We then randomly select 50 samples for each author to compose the training set, keeping all the remaining samples as test instances. We thus obtain 1,000 training samples and 3,048 test samples in total. Figure 4.2 reports the total number of words per author in the training set, divided by political wing, while Figure 4.3 reports the distribution of authors by gender, age,<sup>18</sup> and political party.

<sup>17</sup>Regionalist parties aim for more political power for regional entities.

<sup>18</sup>Note that we use the decade of birth as representation of age group. We assign the closest decade label to each author’s birth; for example, an author born in 1984 is assigned the label “1980”, while an author born in 1987 is assigned the label “1990”.

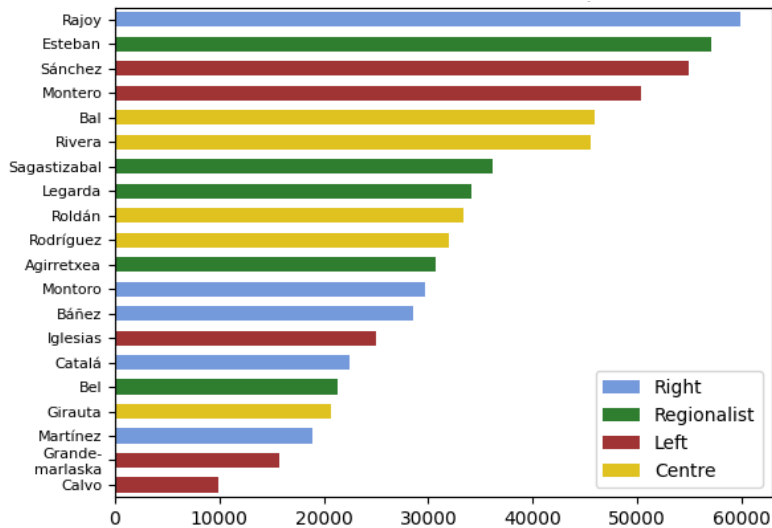


Figure 4.2: Total number of words for each speaker in the training set, grouped by political wing.

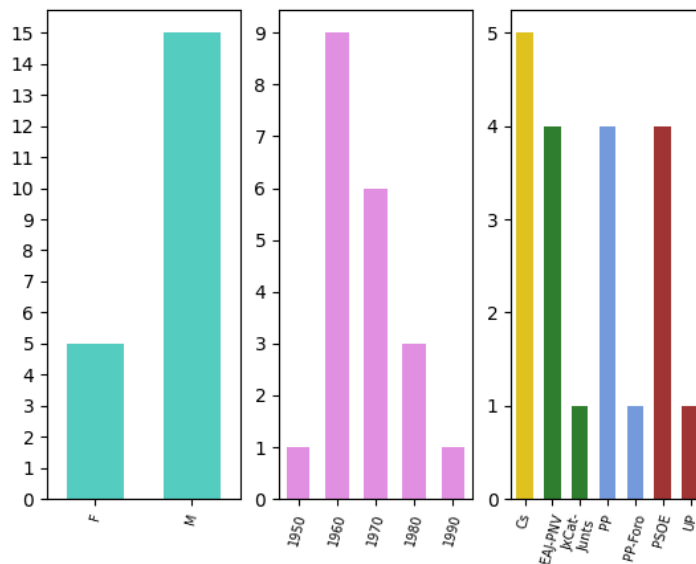


Figure 4.3: Number of speakers for each category by gender, age, and political party.

#### 4.2.3.2 Topic-agnostic features: Base features and text encodings

Following the same approach of the work presented in Section 4.1.3.2, we create both a set with only topic-agnostic features, and different text encodings that do not encase the topical content of the documents.

- **Base features.** As a starting point, we employ a feature set comprised of features routinely used in the AAn field. We again call this set BaseFeatures (from now on: BFs); it is largely based on the same features types described in Section 4.1.3.2. In particular, this set is composed of the following feature types:

- **Function words:** the relative frequency of each function word; we employ the list provided by the NLTK library (see Appendix D);
- **Word lengths:** the relative frequency of each word length, from a minimum of 1 up to a maximum of  $n$  characters, where  $n$  is the maximum word length appearing at least 5 times in the training set;

- **Sentence lengths:** the relative frequency of each sentence length, from a minimum of 1 up to a maximum of  $n$  individual words, where  $n$  is the maximum sentence length appearing at least 5 times in the training set.

For each feature type we compute a matrix  $f \times t$ , where  $f$  is the number of samples in the set and  $t$  is the number of features of the specific type, and we further scale each vector to unit norm (see Footnote 5 in Section 3.3.2). Given the three resulting matrices, we concatenate them in a single final matrix. Note that the dimension of this final matrix varies depending on the maximum word length and sentence length in the dataset, and especially in the training set.

- **Text encodings.** We experiment with various text-encoding approaches, which aim to bring forth some linguistic information that can be employed for AAn, while at the same time keeping the information regarding the topic hidden. In particular, we use five text encoding (from now on: TE) approaches, three of which are based on the LIWC dictionary (see Section 4.2.2).

Note that, since we exclusively use a SVM as our main learning algorithm (see Section 4.2.3.3), these encodings are fed to the algorithm exclusively after a feature extraction process. In fact, for each encoding, we extract the word or character  $n$ -grams from the corresponding output and compute the TfIdf (see Appendix A.3) weights, that we ultimately use as features for the learning algorithm. The resulting feature sets are:

- **TE\_POS:** we replace each word in the document with the respective Part-of-Speech tag (we exploit the POS annotation already available in the ParlaMint dataset); from the encoded text, we then extract the word  $n$ -grams in the range  $[1 \dots 3]$ ;
- **TE\_STRESS:** we extract the character  $n$ -grams in the range  $[1 \dots 7]$  from the encoding of the SS described in Section 4.2.2;
- **TE\_LIWC\_a | TE\_LIWC\_b | TE\_LIWC\_c:** we extract the word  $n$ -grams in the range  $[1 \dots 3]$  from each encoding described in Section 4.2.2.

#### 4.2.3.3 Experimental protocol

From the one hand, we perform AId experiments in two settings: AV for each author of the dataset (where each test sample is labelled as belonging to that author, or not) and AA (where each sample is labelled as belonging to one of the 20 authors). From the other hand, we perform AP experiments by labelling each sample based on the gender, age group, political wing or political party of the author it belongs to.

As evaluation measure, for the AV task we use the well-known  $F_1$  function, and for the AA and AP tasks we use the macro-averaged  $F_1^M$  and micro-averaged  $F_1^\mu$  variants (see Appendix A.1).

We employ SVM as learner;<sup>19</sup> the implementation we employ in this study is the `SVC` module from the `scikit-learn` package (see Appendix D). We perform the optimisation of various hyper-parameters: the parameter  $C$ , which sets the trade-off between the training error and the margin (we explore the range of values in the log-space  $\{10^i\}_{i=-3}^3$ ), the kernel function (we explore the following functions: *linear*, *poly*, *rbf*, *sigmoid*), and whether the classes weights should be balanced or not. The optimization is computed in a grid-search fashion, via 5-fold cross-validation (5-CV) on the training set. The best model is then retrained on the whole training set and is used to make predictions on the test samples.

<sup>19</sup>We also carried out preliminary experiments with Random Forest (RF) and LR; SVM showed a remarkably better performance than RF, while no significant differences were noticed between SVM and LR.

Since the LIWC encodings tend to produce a very high dimensional space,<sup>20</sup> we apply a feature selection approach. We keep only the 10% most important features (employing Chi-square as score measure of importance, see Appendix A.2) for each feature set derived from the LIWC encodings. Note that the selection is always carried out in the training set; during the 5-CV optimization phase, feature selection is carried out in the corresponding 80% of the training set used as training.

As stated, we assess the effect of the different feature types by evaluating the performance of a classifier fed with or without them, in a series of addition and ablation experiments. In the first batch of results, we show the performance of the different feature sets by using the BFs set as baseline, and then adding each of the other feature sets separately. In the second batch of results, we report the performance of the model after subtracting each of the feature sets separately from the combination of all the feature sets (named ALL).

Finally, we also compare the results obtained with the aforementioned features with the results obtained by a method trained on the original text (hence, potentially mining topic-related patterns). To this aim, we employ the pre-trained transformer named BETO-cased [66],<sup>21</sup> from the Huggingface library (see Appendix D). This model obtained better results than the “uncased” version in preliminary experiments. We fine-tune the model for 50 epochs on the training set, with the learning rate set to  $10^{-6}$  and the other hyper-parameters set as default.

In order to better evaluate the difference among the methods, we perform the McNemar’s paired non-parametric statistical hypothesis test [200] between the results obtained using our best SVM configuration and the results obtained using the BETO model, for each of the AAn tasks. The test is carried out by converting the predictions of the two methods into binary values, where 1 stands for a correct prediction and 0 stands for a wrong prediction. We take 0.05 as confidence level.

#### 4.2.4 Results

We show the results of the AV experiments in Table 4.6. BETO obtains the best result in 10 out of 20 cases, 5 of which are statistically significant; conversely, the SVM classifier obtains the best performance in 10 out of 20 cases, 7 of which are statistically significant. Thus, we might consider the performance of the two methods comparable, even though SVM does not exploit any topic-related information (as the BETO transformer instead could do). Focusing on the SVM results, we observe that the best-performing feature set is often (in 10 out of 20 cases) the one combining BFs and TE\_POS, confirming that the syntactic encoding is a good indicator of style. The other best-resulting feature sets are mostly different “ablations” of the ALL set. In particular, it seems that the TE\_LIWC\_c features are rather detrimental, since the configuration ALL - TE\_LIWC\_c yields the best results in 5 cases. Interestingly, we observe severe fluctuations in performance across the authors, with the best result and relative merits of each of the feature sets being strongly dependent on the author under consideration; for example, the feature set TE\_STRESS appears to be beneficial for authors like Agirretxea and Bel, while the same feature set seems to be detrimental for authors like Sagastizabal, Grande-Marlaska, and Montero. We further analyze these results in Section 4.2.5.

We show the results of the AA experiments in Table 4.7. In these experiments, the ALL - TE\_STRESS and the ALL - TE\_LIWC\_c feature combinations employing the SVM learner obtain the best results, both outperforming BETO in a statistically significant sense. In fact, the feature sets TE\_STRESS and TE\_LIWC\_c exhibit a constant disturbance effect.

We show the results of the AP experiments in Table 4.8. While BETO and SVM do not show

<sup>20</sup>Indeed, TE\_LIWC\_a, TE\_LIWC\_b and TE\_LIWC\_c create the highest number of features in our experiments, ranging from 3,000 to more than 20,000.

<sup>21</sup>Available at: <https://huggingface.co/dccuchile/bert-base-spanish-wwm-cased>.

Table 4.6: Results of the AV experiments. The best result for SVM is in **bold**, while the best overall result is in *italic*. The results of the McNemar statistical significance test (**M**) against the baseline are shown for the best SVM result.

	Martinez	M	Sagastizabal	M	Rodríguez	M	Legarda	M	Agirretxea	M	Girauta	M	Esteban	M	Rajoy	M	Sánchez	M	Catalá	M
BFs	.616		.626		.272		.580		.277		.100		.510		.420		.528		.397	
+ TE_POS	<b>.742</b>	*	.761		.524		.672		.560		<b>.188</b>	*	<b>.516</b>	*	<b>.515</b>	*	<b>.640</b>	*	<b>.537</b>	*
+ TE_STRESS	.675		.618		.293		.586		.359		.086		.458		.414		.464		.373	
+ TE_LIWC_a	.529		.517		.091		.621		.329		.060		.365		.165		.535		.277	
+ TE_LIWC_b	.538		.503		.092		.640		.374		.070		.281		.259		.508		.367	
+ TE_LIWC_c	.549		.408		.089		.521		.277		.051		.273		.229		.425		.214	
ALL	.706		.646		.371		.650		.589		.081		.338		.492		.618		.503	
-BFs	.724		<b>.781</b>		.372		<b>.734</b>		.524		.046		.362		.447		.514		.288	
-TE_POS	.599		.415		.229		.543		.403		.078		.291		.348		.504		.347	
-TE_STRESS	.723		.655		.441		.629		.545		.036		.379		.477		.602		.489	
-TE_LIWC_a	.709		.568		.392		.552		.560		.131		.341		.449		.631		.518	
-TE_LIWC_b	.692		.568		.381		.708		.533		.088		.348		.469		.625		.441	
-TE_LIWC_c	.710		.611		<b>.526</b>	*	.572		<b>.674</b>	*	.056		.363		.513		<b>.640</b>	*	.487	
Beto_base_cased	.836		.798		.314		.771		.632		<b>.247</b>		.352		<b>.757</b>		.388		<b>.729</b>	
	Montoro	M	Báñez	M	Iglesias	M	Rivera	M	Roldán	M	Bel	M	Bal	M	Calvo	M	G.Marlasca	M	Montero	M
BFs	.468		.671		<b>.482</b>	*	.526		.161		.599		.203		.227		.362		.479	
+ TE_POS	<b>.540</b>		.717		.444		<b>.654</b>	*	.400		.609		<b>.449</b>	*	<b>.481</b>	*	.425		.453	
+ TE_STRESS	.488		.645		.389		.557		.136		.662		.171		.204		.321		.381	
+ TE_LIWC_a	.379		.539		.319		.443		.080		.460		.047		.189		.295		.431	
+ TE_LIWC_b	.423		.603		.313		.323		.091		.506		.205		.167		.290		.456	
+ TE_LIWC_c	.366		.543		.345		.402		.138		.483		.068		.131		.327		.359	
ALL	.395		<b>.748</b>		.437		.622		.289		.677		.189		.383		.478		.401	
-BFs	.328		.719		.307		.534		.349		.594		.206		.318		.423		.305	
-TE_POS	.356		.657		.403		.463		.237		.568		.090		.209		.409		.346	
-TE_STRESS	.409		.745		.437		.636		.310		.654		.189		.394		<b>.480</b>		.447	
-TE_LIWC_a	.454		.689		.418		.604		.212		<b>.703</b>	*	.200		.268		.420		.501	
-TE_LIWC_b	.456		.707		.421		.587		.184		.652		.267		.329		.386		.384	
-TE_LIWC_c	.435		.747		.454		.629		<b>.471</b>		.683		.175		.353		.408		<b>.570</b>	*
Beto_base_cased	.610		<b>.800</b>		.437		.460		.468		.601		.381		<b>.494</b>		<b>.664</b>		.426	

Table 4.7: Results for the AA experiment; we employ the same notation as in Table 4.6.

	AA		
	$F_1^M$	$F_1^\mu$	M
BFs	.401	.444	
+ POS	.570	.620	
+ STRESS	.392	.436	
+ LIWC_a	.430	.480	
+ LIWC_b	.446	.493	
+ LIWC_c	.348	.394	
ALL	.580	.631	
- BFs	.545	.599	
- POS	.435	.485	
- STRESS	<b>.585</b>	<b>.638</b>	*
- LIWC_a	.565	.615	
- LIWC_b	.562	.613	
- LIWC_c	<b>.585</b>	.635	*
Beto_base_cased	.417	.471	

Table 4.8: Results for the AP experiments; we employ the same notation as in Table 4.6.

	Gender			Age			Wing			Party		
	$F_1^M$	$F_1^\mu$	M	$F_1^M$	$F_1^\mu$	M	$F_1^M$	$F_1^\mu$	M	$F_1^M$	$F_1^\mu$	M
BFs	.720	.802		.428	.478		.599	.631		.547	.563	
+ TE_POS	.751	.828		.545	.592		.685	.715		<b>.642</b>	<b>.681</b>	
+ TE_STRESS	.705	.803		.402	.459		.581	.613		.539	.561	
+ TE_LIWC_a	.696	.788		.421	.469		.601	.636		.508	.536	
+ TE_LIWC_b	.716	.812		.441	.511		.619	.653		.492	.502	
+ TE_LIWC_c	.669	.777		.366	.434		.525	.554		.492	.531	
ALL	.736	<b>.854</b>		.551	.589		.690	.719		.604	.648	
- BFs	.709	.843		.485	.540		.650	.681		.552	.614	
- TE_POS	.700	.825		.411	.469		.640	.669		.498	.544	
- TE_STRESS	.732	.850		.553	.594		<b>.707</b>	.736	*	.610	.658	
- TE_LIWC_a	<b>.754</b>	<b>.854</b>		.522	.573		.677	.709		.611	.637	
- TE_LIWC_b	.736	.838		.522	.564		.687	.717		.610	.637	
- TE_LIWC_c	.725	.831		<b>.573</b>	<b>.609</b>	*	.706	<b>.737</b>	*	.613	.650	
Beto_base_cased	.762	.847		.337	.420		.666	.698		.574	.662	

remarkable differences for gender prediction, SVM excels in the other tasks, with statistical significance in the case of age and political wing. Consistently with what observed for the AA and AV cases, the TE\_POS feature set shows a clear proficiency, while the contrary can be said for TE\_LIWC\_c.

Overall, these experiments allow us to draw some interesting conclusions regarding the features we study for AAn in the Spanish language: on the one hand, TfIdf-weighted  $n$ -grams computed on POS-tags encodings are effective for multiple tasks and settings; on the other hand, the feature sets TE\_STRESS and TE\_LIWC\_c tend to fare poorly. Interestingly enough, the combination of multiple topic-agnostic feature sets proved to fare comparably to, and to outperform in some cases, a state-of-the-art neural network that has full access to topic-related information.

#### 4.2.5 Post-hoc analysis of the AV results

Given the differences in performance spotted in the AV results among the authors of the dataset (see Section 4.1.4, Table 4.6), we see analogous differences in performance in the AV tasks regarding the *Epistle to Cangrande* (see Section 3.4, Table 3.3). We further analyse the system behaviour in order to outline a suitable explanation for such variances.

We hypothesize that these differences might be associated with the authors' personal characteristics

(political affiliation, gender or age), or communication style. In order to evaluate these hypotheses, we resort to a series of tools for data analysis: the one-way ANOVA test (Section 4.2.5.1) and the Spearman test applied to various communication indices (Section 4.2.5.2).

#### 4.2.5.1 One-way ANOVA test for political groups

In this section, we aim to test if, by grouping the speakers by categories (political wing, political party, gender, or age), statistically significant differences in performance emerge from the various sets of features in the AV task. To this aim, we employ the one-way ANOVA test, which is a parametric test used to check for statistically significant differences in any outcome among groups that are under one categorical variable. We use 0.05 as confidence level, and we check that the assumptions for the test (independence, normality and homogeneity of variance) are met. Note that we only consider groups with more than one member, e.g., when grouping by age, we do not consider the groups corresponding to decades 1950 and 1990, since each group would have only one member, Montoro and Rodríguez, respectively.

With this analysis, we do not find any significant difference employing the grouping by gender or by age. However, we indeed find that the  $F_1$  results display significant differences for multiple feature sets and for BETO if grouped by political party, and especially so if grouped by political wing. The results are reported in Table 4.9.

Tukey’s Honestly Significant Difference (HSD) [285], a statistical test that compares all possible pairs of means among various result groups, applied to the AV results, reveals that:

- when grouping by political wing, the significant difference always occurs between the Centre and the other wings;
- when grouping by political party, the significant difference occurs between Cs and EAJ-PNV, and between Cs and PP.

If we focus our attention to the features BFs + TE\_LIWC\_b (the only SVM feature setting giving rise to statistically significant differences in performance when the groups are generated by wing, and also when the groups are generated by political party), it turns out that authors belonging to the Centre/Cs obtain  $F_1$  scores significantly lower than authors from other groups (see Figure 4.4). As a possible explanation, we observe that the Cs is a relatively new political party (it was founded in 2006), and its members have been in various different parties before joining it; this could have lead to a certain difficulty in creating a specific personal style.

#### 4.2.5.2 Spearman coefficient applied to style indices

In order to assess whether the different communication style might impact the results of the classifiers in the AV experiments, we employ three indices created to measure different aspects of the communication, based on LIWC categories. The indices are:

- **ATI**: the Analytic Thinking Index (ATI), introduced by Pennebaker et al. [227] and formerly named Categorical Dynamic Index, is a unit-weighted score computed from grammatical categories derived from LIWC. This measure is based on the observation that the use of articles and prepositions is associated with a more abstract thinking, while the use of pronouns, auxiliary verbs, conjunctions, adverbs, and negations is associated with a more intuitive and narrative style. The index considers the following grammatical items, and in particular the respective percentage over the total number of words (we report the notation in parentheses): articles ( $a$ ), prepositions ( $p$ ), pronouns ( $r$ ), auxiliary verbs ( $x$ ), conjunctions ( $c$ ), adverbs ( $d$ ) and negations ( $n$ ). Thus, ATI is computed as:

$$ATI = a + p - r - x - c - d - n \quad (4.1)$$

Table 4.10: Spearman correlation values ( $r$ ) and p-values ( $p$ ) between the  $F_1$  values in the AV experiments and communication indices values; statistically significant values are in bold. ANOVA p-values on the  $F_1$  results on the AV experiments when grouped by wing and by party; statistically significant values are in bold.

	Wing	Party	ATI		CNI		ASI	
			$r$	$p$	$r$	$p$	$r$	$p$
BFs	<b>.026</b>	.085	.335	.148	.432	.057	-.444	.050
+ TE_POS	.078	.192	.296	.205	<b>.460</b>	<b>.041</b>	<b>-.683</b>	<b>.001</b>
+ TE_STRESS	<b>.014</b>	.060	.284	.225	.411	.072	<b>-.508</b>	<b>.022</b>
+ TE_LIWC_a	<b>.023</b>	.054	.403	.078	<b>.558</b>	<b>.011</b>	-.429	.059
+ TE_LIWC_b	<b>.008</b>	<b>.031</b>	<b>.489</b>	<b>.029</b>	<b>.621</b>	<b>.003</b>	<b>-.489</b>	<b>.029</b>
+ TE_LIWC_c	<b>.042</b>	.117	.438	.054	<b>.543</b>	<b>.013</b>	-.400	.081
ALL	<b>.045</b>	.136	.335	.148	<b>.483</b>	<b>.031</b>	<b>-.459</b>	<b>.042</b>
- BFs	.065	.135	.251	.286	.389	.090	<b>-.642</b>	<b>.002</b>
- TE_POS	<b>.045</b>	.133	.360	.119	<b>.471</b>	<b>.036</b>	-.402	.079
- TE_STRESS	.071	.185	.314	.177	<b>.463</b>	<b>.040</b>	<b>-.487</b>	<b>.029</b>
- TE_LIWC_a	.054	.159	.271	.248	.442	.051	-.435	.056
- TE_LIWC_b	<b>.042</b>	.136	.319	.171	<b>.453</b>	<b>.045</b>	<b>-.543</b>	<b>.013</b>
- TE_LIWC_c	.192	.399	.211	.373	.331	.154	<b>-.493</b>	<b>.027</b>
Beto_base_cased	<b>.001</b>	<b>.008</b>	<b>.544</b>	<b>.013</b>	<b>.675</b>	<b>.001</b>	<b>-.517</b>	<b>.020</b>

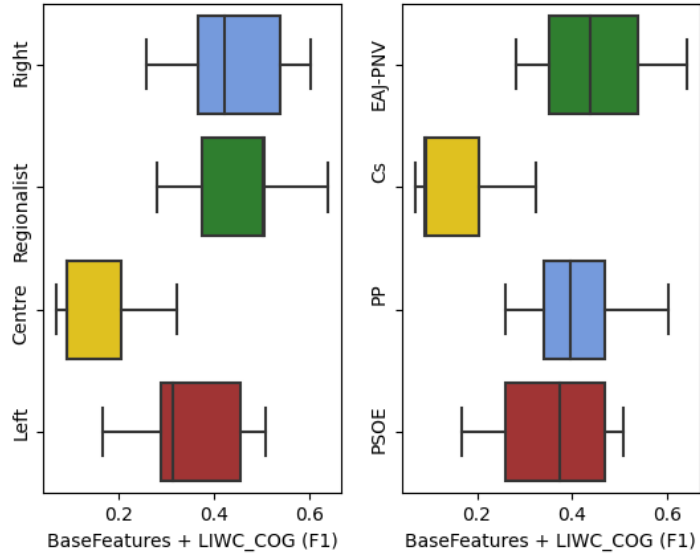


Figure 4.4: AV results for the BF + TE\_LIWC\_b feature set, divided by political wing and party.

A positive score denotes a more analytic thinking, while a negative score denotes a more intuitive one. This score has already been used to analyze long-term trends in political language in EEUU [150].

- **CNI**: the Categorical-versus-Narrative Index (CNI), introduced by Ortega-Bueno et al. [220] and inspired by the study by Nisbett et al. [217], is similar in nature to ATI. The index considers the following grammatical items, and in particular the respective percentage over the total number of words (we report the notation in parentheses): nouns ( $n$ ), adjectives ( $j$ ), prepositions ( $p$ ), verbs ( $v$ ), adverbs ( $d$ ) and personal pronouns ( $o$ ). Thus, CNI is computed as:

$$\text{CNI} = n + j + p - v - d - o \quad (4.2)$$

Like ATI, a higher score of CNI denotes a language more focused on the exposition of abstract concepts, while a lower score denotes a language more prone to narration and storytelling.

- **ASI**: the Adversarial Style Index (ASI), first proposed by Chulvi et al. [73], is a ratio representing how much an author refers directly to the political adversary in a confronting manner in political debates. Adversarial speech has been vastly studied in parliamentary and election debates, both in the English [58] and Spanish context [46]. The index considers the following grammatical items, and in particular the respective percentage over the total number of words (we report the notation in parentheses): singular and plural first-person pronouns and verbs ( $y$ ), and singular and plural second-person pronouns and verbs ( $u$ ). Thus, ASI is computed as:

$$\text{ASI} = \frac{u}{y + u} \quad (4.3)$$

We show the ATI, CNI and ASI scores computed for each author on the entire dataset in Figure 4.5, Figure 4.6 and Figure 4.7, respectively.

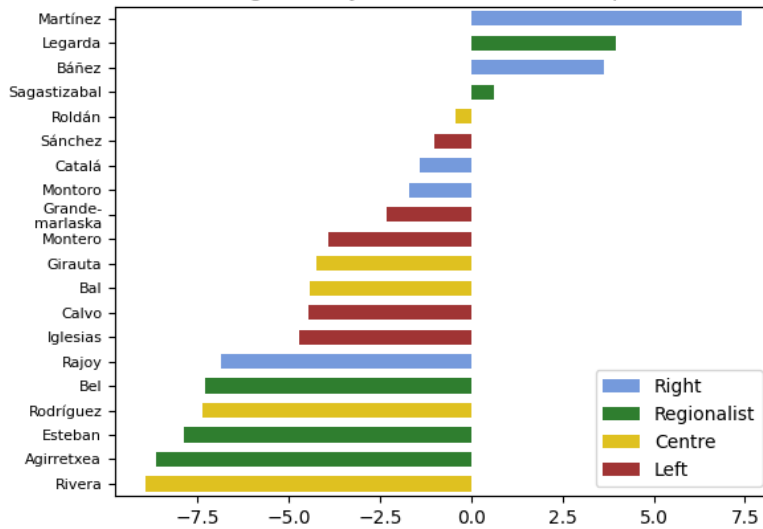


Figure 4.5: Analytical Thinking Index values for each author.

We employ these measures to quantify the extent to which the AV performance correlates to certain styles of communication. To this aim, in Table 4.10 we show, for each of the indices, the Spearman correlation coefficient ( $r$ ) between the classification scores and the authors' index scores.

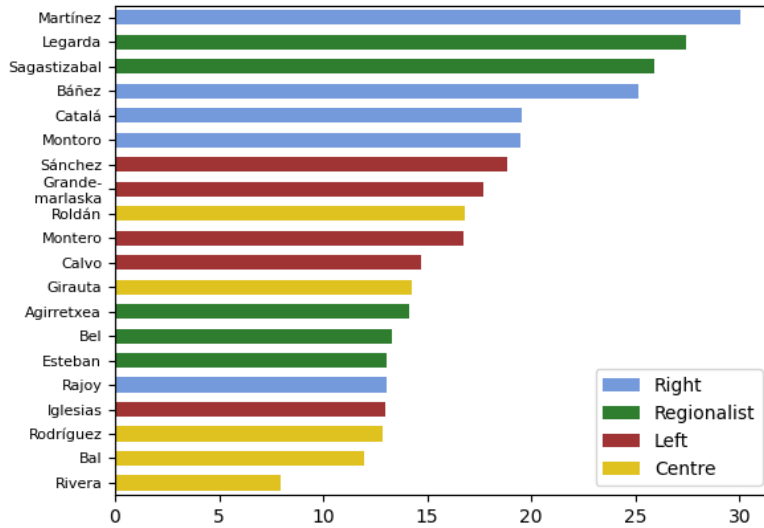


Figure 4.6: Categorical-versus-Narrative Index values for each author.

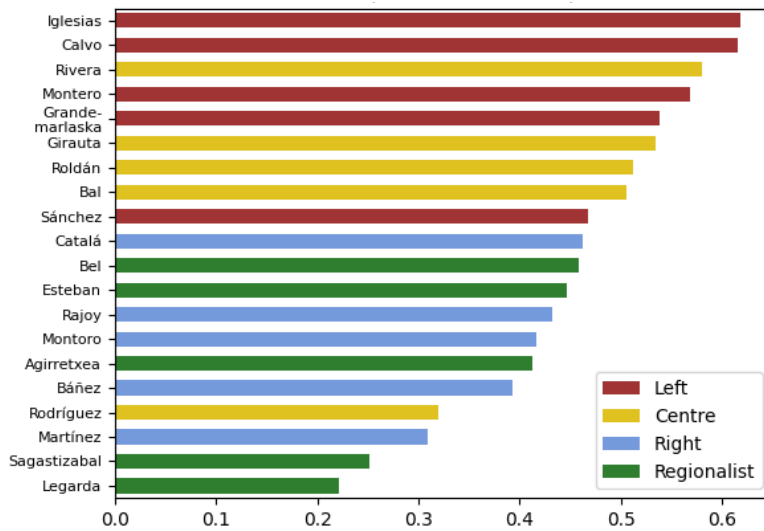


Figure 4.7: Adversarial Style Index values for each author.

We see that the BETO model displays the strongest positive correlation with respect to ATI and CNI, followed closely by the BFs+ TE\_LIWC\_b feature set. This seems reasonable, since ATI and CNI hinge upon abstract explanations and concepts, which are captured by TE\_LIWC\_b, while a big portion of the training of BETO is comprised of sources like Wikipedia, legislative texts and talks.

Interestingly, the correlations between  $F_1$  scores and the psycholinguistic indexes obtained by each author is found to be statistically significant for more feature sets combinations for CNI (8) than for ATI (1). We hypothesize that this might be due to the fact that ATI, unlike CNI, captures a degree of formality that is rather common in Parliamentary speeches, hence preventing meaningful differences to emerge.

Moreover, while 8 feature sets show a positive correlation with CNI, as many feature sets show a negative correlation with ASI, with BFs + TE\_POS holding the strongest correlation. For comparison, we show the plot of both CNI and ASI correlated with the results for the ALL feature set in Figure 4.8 and Figure 4.9 respectively. The opposite nature of the two indices is understandable, since a more adversarial style would naturally be less abstract and more focused on events and narration. Indeed, some studies,

both regarding Question Time in English [97, 124] and face-to-face Spanish political debates [46], noted that adversarial speeches in the Parliamentary context present certain repeating oratory patterns. This could explain why the present features, and in particular the TE\_POS set, perform worse on speakers with higher ASI, who are likely to use common syntactic patterns. Conversely, it is easier to recognize speakers with a more abstract communication style.

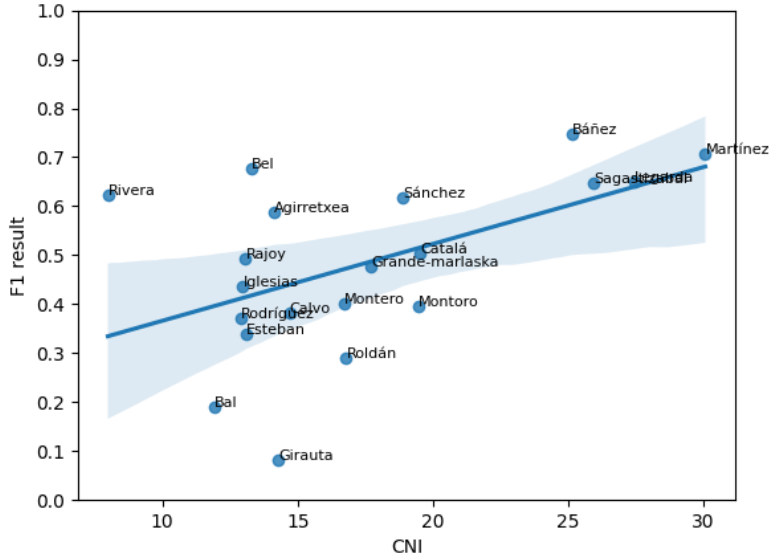


Figure 4.8: Correlation among AV results for the ALL feature set and CNI values.

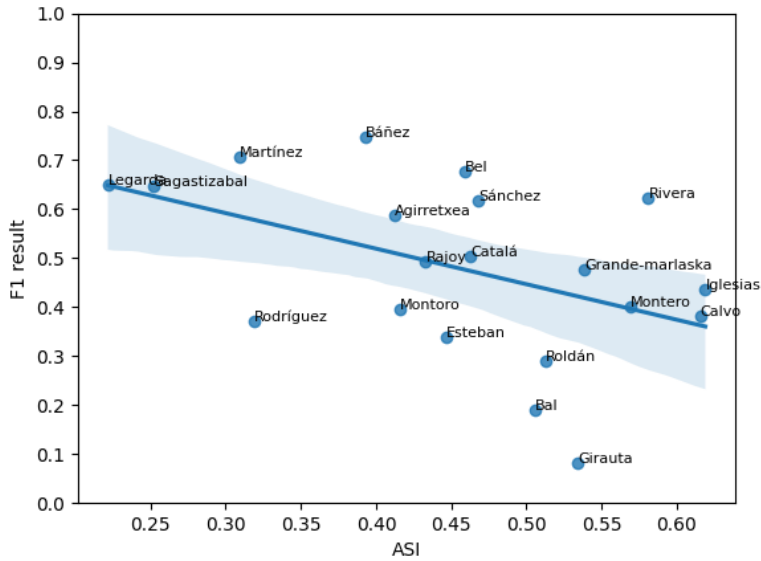


Figure 4.9: Correlation among AV results for the ALL feature set and ASI values.

### 4.3 Discussion

In this chapter, we have tackled what in this Thesis we define as **Problem 1**, which is the problem of defining a feature type for AAn that would be effective for the authorship tasks, and at the same time would not carry information regarding the topic of the documents. In this regard, we proposed to employ the rhythm of the discourse, and we reported detailed experiments on the subject.

In particular, in Section 4.1, we exploited the notion of “syllabic quantity” in order to compute rhythmic features that we tested in the AA task of Latin prose texts. Indeed, these SQ-based features show a beneficial effect in the majority of cases, at least while employing SVM as the learning algorithm. The contribution of SQ-based features is instead uncertain while employing a NN method, since the difference in performance is rarely significant. We conjecture that this might be due to the limited datasets size, an issue that severely restrict the use of NN architecture in cultural heritage settings, as we have seen in Section 2.4.

Given the promising results in the Latin setting, in Section 4.2 we extended the research of rhythmic features to the Spanish language and to a wider range of AAn tasks (AA, AV and various AP tasks), this time exploiting the concept of “syllabic stress”. We also investigated the effect on performance of three other topic-agnostic feature types, based on psycholinguistics. Indeed, we showed that the combination of such topic-agnostic features performs comparably to a transformer model potentially learning from topic-related information in all such tasks. Moreover, additional experiments seem to suggest that these results are at least partially linked with the political affiliation of the authors and their communication style.

As a conclusion of this chapter, we note that rhythmic- and psycholinguistics-based features, despite not being always beneficial, show a strong potential for AAn applications, and should therefore be taken into consideration when devising an authorship system (e.g., included as part of the model selection phase). More in general, we can affirm that employing topic-agnostic systems for AAn is not only a preferable option from a methodological point of view, but can still lead to performance results which are on par with state-of-art topic-aware methods. This indeed proves the importance of prosecuting the research regarding the features to employ in the AAn field.

In future work, it would be interesting to better analyze the settings in which the features we have here presented are more (or less) effective. For example, given that the SQ-based features have proven effective in Latin prose texts, it would be interesting to evaluate their effect in Latin theatrical pieces, a literary genre that lies in between poetry and prose, and is rich of texts of declamatory nature. Similarly, it would be worthwhile to assess the effect of psycholinguistic features in other, less formal forms of political communications than parliamentary debates, such as tweets [104], where people might be more open to express emotional content.



## Chapter 5

# Diff-vectors as an alternative representation for limited-sized datasets

As explained in Chapter 2, in most AAn tasks textual documents are represented as numerical vectors, following the Vector Space Model. In particular, as in many supervised learning endeavours, each document is represented as a vector of features, where the value of a feature in a vector usually corresponds to a certain linguistic event or textual trait as occurring within the corresponding document.

Koppel and Winter [176] describe an alternative method for generating vectorial representations of texts for AId. Specifically, while in the “standard” representation methodology a vector represents a document, in this alternative method a vector represents an unordered pair of different documents; while in the “standard” methodology the value of a feature is the numerical value of a given linguistic phenomenon occurring in the document, in this alternative method it is the absolute difference between the values of this phenomenon occurring in the two documents. Since these vectors represent differences, we call this representation *Diff-Vectors* (from now on: DVs). While in the standard methodology the class label is the author of the document, in this DV-based methodology the class label is one of the two classes Same or Different (standing for “same author” or “different authors”, respectively), as in the SAV task.

However, the goal of Koppel and Winter [176] was to propose a different method (the IM for SAV, that we already described in Section 2.2), and not to propose the DV-based methodology, which they dismiss as a “simplistic baseline method” [176, p. 179]. Since then, the use of DVs has never been studied systematically.

We argue that this representation might help mitigate the problem of datasets with extremely limited size, a problem that we define as **Problem 2** in this Thesis, and that is very common in the cultural heritage setting. In fact, given  $n$  labelled documents, while the standard methodology gives rise to  $n$  training vectors, the DV-based methodology gives rise to  $O(n^2)$  training vectors, which seems, at first sight, advantageous.

In this chapter, we further explore this intuition, and we carry out extensive experiments in order to determine whether using DVs in place of “standard” vectors brings about higher accuracy in AId tasks. In particular, after presenting some related work (Section 5.1), in Section 5.2 we formally describe DVs and justify why they appear as a superior means of representing authorship-related information. In Section 5.3, we show how to tackle the three AId tasks by means of DVs: SAV (for which DVs are naturally designed for), AA and AV (for which we propose two methods employing a DV-based SAV classifier as a building block). In Section 5.4 we describe our experimental setting, while Section 5.5 reports the results of our experiments. Finally, Section 5.6 wraps up, also pointing at avenues for further

research.

The research presented in this chapter is published in the article by Corbara et al. [5].<sup>1</sup>

## 5.1 Related work

The concept of vectors representing pairs of items is not new in the field of classification as a whole. Just to give some examples, Guo et al. [115] propose to learn a classifier such that, given a pair of input patterns  $x_i$  and  $x_j$ , it computes the conditional joint distribution over the pair of labels  $y_i$  and  $y_j$ ; the assignment of the single label is coordinated (hence the name of the method, Coordination Classifier) by propagating beliefs on a graph over the data. Dumpala et al. [88], tackling the problem of class imbalance, propose a similar method to represent and process the input, while the output label is obtained through majority voting. Both these works underline the gain in the number of samples, and the ensemble mechanism obtained via a single base classifier.

As previously mentioned, the work by Koppel and Winter [176] was the first to present vectors representing pairs of documents in the AAn literature. In this representation, a vector represents two documents, the label of the vector is either **Same** or **Different**, character 4-grams are used as features, and the value of each feature is the absolute difference between the TfIdf weights of the feature in the two documents. As already explained, the goal of Koppel and Winter [176] was to propose a different method (the IM method for SAV, that we already described in Section 2.2), and they dismiss the DV-based representation as a “simplistic baseline method” [176, p. 179].

Since then, a number of authors started to view the AV task in terms of predicting whether vector  $f(D_{A^*}, d_u)$  belongs to class **Same** or to class **Different**, where  $f(D_{A^*}, d_u)$  is a vector derived from the entire set (here represented as  $D_{A^*}$ ) of training documents known to be by candidate author  $A^*$ , and from the document of unknown paternity (here noted as  $d_u$ ). For instance, in Bartoli et al. [31] vector  $f(D_{A^*}, d_u)$  is a vector in which each feature value is the absolute difference between the value of the feature in  $d_u$  and the mean of the values of the feature across the documents in  $D_{A^*}$ . In their well-known Unmasking approach, Koppel et al. [175] split the documents composing  $D_{A^*}$  and  $d_u$  and try to iteratively classify the various splits, removing the most representative features at each run and hence creating a decreasing accuracy curve (the steeper the curve, the more likely the author of  $D_{A^*}$  is taken to be the author of  $d_u$ ; see also the explanation in Section 2.3). Boenninghoff et al. [49] feed the tuple  $(D_{A^*}, d_u)$  to a Hierarchical Recurrent Siamese Network (HRSN), where  $D_{A^*}$  is the concatenation of all the documents from the author considered, in order to analyse the similarity between the two objects. Note that in these methods it is necessary to convert  $D_{A^*}$ , i.e., the available production of the author of interest, into an “artificial” object in order to be able to compare it with the only unknown document  $d_u$ , whether by arbitrarily splitting the documents, by extrapolating single agglomerated measures from their individual characteristics (e.g., the average of the feature values, as in Bartoli et al. [31]), or by combining them into a single entity. Conversely, our method does not require this step, and preserves the integrity of the document.

A slightly different approach is used in the PRNN method presented by Hosseinia and Mukherjee [134]. They view  $D_{A^*}$  as a single document (generated by the concatenation of all the documents in it), and they use both this document and document  $d_u$  as input for a parallel neural network composed of an embedding layer and a RNN layer, finally combining the two outputs by computing a vector  $f(D_{A^*}, d_u)$  consisting of values of similarity between the two documents. The same work also proposes another method, called TE, which is based on a transformation encoder that transforms the vector representing

---

<sup>1</sup>The code to replicate the experiments is available at: <https://github.com/AlexMoreo/diff-vectors>.

$A^*$  into the vector representing  $d_u$ , and takes the resulting loss as a measure of similarity; the authors repeat the process several times using different feature sets, and generate a vector  $f(D_{A^*}, d_u)$  consisting of the different similarity values. In a similar vein, in Bevendorff et al. [40] the  $f(D_{A^*}, d_u)$  vector is composed of 7 similarity values computed on the char  $n$ -grams of the two documents. Unlike the present work, none of the above works attempts to tackle the AV and AA tasks by recasting them in terms of SAV.

More generally, pairs in the form of documents  $(d_j, d_u)$  are employed also for AV and AA tasks within approaches based on similarity (or dissimilarity) measures; besides the well-known IM method by Koppel and Winter [176] with its many variants (see Section 2.2), an example of this is the method by Jankowska et al. [142]. Along this vein, Cerra et al. [68] and Halvani et al. [120] mix this concept with compression algorithms, employing the Fast Compression Distance and the Compression-Based Cosine, respectively. Note that these methods, when applied to AV or AA tasks, often employ something akin to a KNN scheme, assigning the unknown document to the author whose texts are most similar to it, according to the measure of choice. The differences between these methods and the one we discuss are evident, since the former do not train a learner; additionally, and as a direct consequence, our KNN-inspired approach (see Section 5.3.2.1) is based on posterior probabilities.

Recently, Ikae [138], Menta and Garcia-Serrano [202], Weerasinghe et al. [293] tested the use of DVs for the open-set SAV problem at the PAN-2021 shared task; in particular, Menta and Garcia-Serrano [202] propose a method that feeds DVs to a double-channel NN, where the feature values are the TfIdf weights of character  $n$ -grams in one channel, and of punctuation marks in the other channel. The outputs of the two channels are then concatenated in a final series of layers, that ultimately leads to the classification decision.

Finally, we note that the the DV-based representation that we have discussed is reminiscent of ideas that have been independently explored in multilingual text classification. In particular, Moreo et al. [205] investigate the idea of applying lightweight random projections to the feature space. Mathematically, a random projection  $\mathbf{X}\mathbf{R}$  of a matrix  $\mathbf{X} \in \mathbb{R}^{np}$ , where  $n$  is the number of documents and  $p$  is the number of features, can be attained by multiplying it with a random matrix  $\mathbf{R} \in \mathbb{R}^{pr}$ , where  $r \ll p$  is the number of dimensions. The term “lightweight” refers to the fact that the rows in  $\mathbf{R}$  contain only two non-zero values  $(-1,+1)$ . The pair-based version  $\mathcal{L}_{\mathcal{P}}$  of a dataset  $\mathcal{L}$  can be defined in terms of  $|\mathbf{R} \cdot \mathbf{X}|$ , where  $\mathbf{X} \in \mathbb{R}^{np}$  is the document-by-feature matrix and  $\mathbf{R}$  is instead a lightweight projection matrix  $\mathbf{R}^{rn}$ , this time with  $r$ , the number of pairs, much higher than  $n$ ; here  $|\cdot|$  represents the element-wise absolute value. Such a projection effectively computes the absolute difference between two chosen documents. Esuli et al. [91] instead propose *funnelling*, a method relying on a variant of stacking that, similar to our StackedAA technique (see Section 5.3.2.2), learns a meta-classifier on top of the posterior probabilities returned by classifiers working on the input space. Differently from our StackedAA, funnelling computes the posterior probabilities with respect to the target categories, and not to other documents.

## 5.2 Diff-Vectors: Characteristics and advantages in AId applications

As already explained in Chapter 2, in “standard” AAn each document  $d_i$  is usually represented via a labelled vector  $\mathbf{x}_i$  of features, where the value  $\mathbf{x}_i^k$  of the  $k$ -th feature in vector  $\mathbf{x}_i$  usually represents a certain linguistic event or textual trait as it appears in  $d_i$ ; in the case of AId, the label  $y_i \in \mathcal{A}$  represents the true author of  $d_i$ .

In this study, we investigate an alternative type of vectorial representation for AId tasks. Here, a labelled vector  $\mathbf{x}_{ij}$  represents an unordered pair  $(d_i, d_j)$  of documents in  $\mathcal{D}$  such that  $i \neq j$ , each feature

represents a linguistic phenomenon, the label  $y_{ij} \in \mathcal{P} = \{\text{Same}, \text{Different}\}$  indicates whether the true authors of  $d_i$  and  $d_j$  are the same person or not, and the value  $\mathbf{x}_{ij}^k$  of the  $k$ -th feature in vector  $\mathbf{x}_{ij}$  represents the absolute difference between the feature values of the linguistic phenomenon in  $d_i$  and  $d_j$ . Since the *difference* between relative frequencies is central to the definition of these vectors, we call them *Diff-Vectors* (DVs).

If the chosen features are indeed indicative of authorship, when two documents have been written by the same author the values  $\mathbf{x}_{ij}^k$  of these features will be low, since the feature values will be similar in the two documents. In other words, DVs belonging to class **Same** will tend to be characterised by low feature values and low norms, while vectors belonging to class **Different** will tend to be characterised by high feature values and high norms. The quintessential (although fairly improbable) example of a DV in class **Same** is the vector of all 0's; conversely, the quintessential (although fairly improbable) example of a DV in class **Different** is (if feature values are all normalised) a vector of all 1's, since it represents two documents with maximally different values for all features. All DVs fall, if normalised, in the unit hypercube.

Any set of labelled documents  $\mathcal{D} = \{(d_1, y_1), \dots, (d_n, y_n)\}$  can be represented either in the standard way or via DVs. One of the main differences between the two representations is that the standard representation gives rise to  $n$  labelled vectors, while the alternative representation gives rise to  $n(n-1)/2$  labelled vectors. The other main difference is that a classifier using the standard representation attempts to predict, given an unlabelled document, its true author, while a classifier using the DV-based representation attempts to predict, given two unlabelled documents, whether the two documents are by the same author or not. In other words, the standard representation is geared towards AV or AA, while the DV-based representation is geared towards SAV. However, AV and AA can (as discussed in Section 5.3) be recast in terms of SAV, and vice-versa; as a result, we will consider the two representations as general-purpose alternatives, and we will study them as such.

Both these aspects support the intuition that DV-based representation should be advantageous for solving the two tasks of AV and AA, although it not naturally designed for this purpose. We get into more details explaining the theoretical reasoning behind this argument in the following Section 5.2.1 and Section 5.2.2, respectively.

### 5.2.1 More training examples for AV settings

Let us consider the AV setting, and let us assume that  $A^* \in \mathcal{A}$  is our candidate author.

When using the standard representation, we typically replace each label in  $\mathcal{A} \setminus \{A^*\}$  with label  $\overline{A^*}$  (to indicate the complement of  $A^*$ ) and train a binary classifier that discriminates between  $A^*$  and  $\overline{A^*}$ . However, by doing so, the information regarding whether two training examples in  $\overline{A^*}$  are by the same author or not is completely lost. For authorship-related tasks this is valuable information, which the standard representation wastes and the DV-based representation retains. The following example shows that the information wasted by the standard representation is, indeed, a lot.

**Example 1.** *Assume a set of 10 authors and a training set consisting of 100 training examples for each author. The DV-based representation gives rise to  $(1,000 \cdot 999)/2 = 499,500$  DV, among which:*

1.  $10 \cdot (100 \cdot 99)/2 = 49,500$  examples have label **Same**, since for each author  $A_z \in \{A_1, \dots, A_{10}\}$  there are  $(100 \cdot 99)/2 = 4,950$  unordered pairs of different examples such that the author of both examples is  $A_z$ ; of these

(a)  $(100 \cdot 99)/2 = 4,950$  are such that the author of both examples is  $A^*$ ;

(b)  $9 \cdot (100 \cdot 99)/2 = 44,550$  are such that the author of both examples is  $A_z$  for some  $A_z \neq A^*$ ;

2.  $45 \cdot (100 \cdot 100) = 450,000$  examples have label *Different*, since there are  $10 \cdot 9 / 2 = 45$  unordered pairs  $(A', A'')$  of different authors, and for each such pair there are  $100 \cdot 100 = 10,000$  pairs of examples in which one example is by  $A'$  and the other example is by  $A''$ ; of these

(a)  $9 \cdot (100 \cdot 100) = 90,000$  are such that one of  $A'$  and  $A''$  is  $A^*$ ;

(b)  $36 \cdot (100 \cdot 100) = 360,000$  are such that neither of  $A'$  and  $A''$  is  $A^*$ .

Note that the information provided to the training process by the examples of Type-1a is also provided (albeit in a different form) when using the standard representation, since with the latter the learner is implicitly told that the two documents are from the same author. The same happens for the examples of Type-2a, since with the standard representation the learner is implicitly told that the two documents are from different authors. However, the key observation here is that the examples of Type-1b and Type-2b provide information that is instead lost when using the standard representation, since the standard representation only tells the learner that the two documents are not by  $A^*$ , but does not tell the learner if they are by the same author or not. In sum, 404,550 out of 499,500 training examples, i.e., about 81% of the entire set, provide information that was not provided by the standard representation; in other words, in this case the learner receives more than 5 times the amount of information than the standard representation provides to it.  $\square$

More in general, if we have  $m$  authors and  $q = n/m$  training examples per author, the number of DVs that do not provide additional information with respect to the standard representation is:

$$\frac{q(q-1)}{2} + (m-1)q^2 \quad (5.1)$$

i.e., the number of pairs of Type-1a plus the number of pairs of Type-2a. Instead, the number of DVs that do provide additional information is:

$$\frac{(m-1)q(q-1)}{2} + \frac{(m-1)(m-2)q^2}{2} \quad (5.2)$$

i.e., the number of pairs of Type-1b plus the number of pairs of Type-2b. Note that, while the amount of information that was already available to the learning process is  $O(mq^2)$  (Equation 5.1), the new information made available to it is  $O(m^2q^2)$  (Equation 5.2). The latter amount of information can be extremely valuable, especially since it comes at no cost, and especially in application scenarios characterised by the scarcity of training data (as it is rather common in AId, especially in the cultural heritage domain). Among all of the above,

$$\frac{q(q-1)}{2} + \frac{(m-1)q(q-1)}{2} = \frac{mq(q-1)}{2} \quad (5.3)$$

are examples of *Same*, which are  $O(mq^2)$ , while

$$(m-1)q^2 + \frac{(m-1)(m-2)q^2}{2} = \frac{m(m-1)q^2}{2} \quad (5.4)$$

are examples of *Different*, which are  $O(m^2q^2)$ .

In sum, when our task is AV, if we switch from the standard representation to DV-based representation, we end up with a much higher quantity of training data, since DV-based representation exploits

information that the standard representation waste. However, note that switching from standard representations to DV-based representations means switching from vectors geared towards AV to vectors geared towards SAV. This suggests the idea to use these vectors to train a high-performance SAV classifier, and then to devise an algorithm that can perform AV on top of this SAV classifier; this is in fact the aim of Section 5.3.3.

## 5.2.2 More robust training in AA settings

In general, the fact that more information is provided to the training process only holds for tasks in which the training documents by different authors end up being grouped together into a single class, which is exactly the case for AV with the  $\bar{A}^*$  class (see previous Section 5.2.1). This does not necessarily hold for other AId tasks as well.

Indeed, the fact that more information is provided to the training process does not hold when the above-mentioned grouping does not happen, as in closed-set AA. In this task, the information conveyed to the training process by a DV with label **Same** is obviously also implicitly conveyed when using the standard representation (where the vectors corresponding to the two documents are labelled with the same author class), and it happens for DVs with label **Different** (where the two vectors are labelled with different authors classes).

So, in closed-set AA it would appear that there is no advantage in using DVs. This is actually not true, because the advantage is in the fact that, when using DVs, all the training information is concentrated on labelling just two classes, i.e., **Same** and **Different**, while in the standard representation this information is spread out thin, i.e., it is used for labelling  $m$  different classes, each one thus ending up having a smaller number of positive training examples. The following example makes the point more concrete.

**Example 2.** *Assume we are dealing with closed-set AA; assume a set of  $m = 10$  authors and a training set consisting of  $q = 20$  training examples for each author. The standard representation gives rise to  $q \cdot m = 200$  training vectors, 20 for each class, while the DV-based representation gives rise to  $mq(mq - 1)/2 = 19,900$  training vectors, among which  $10 \cdot (20 \cdot 19)/2 = 1,900$  DVs for class **Same** and  $10 \cdot 9 \cdot 20^2/2 = 18,000$  DVs for class **Different**.  $\square$*

More in general, if we have  $m$  authors and  $q$  training examples per author, in closed-set AA we have  $mq(q - 1)/2$  DVs of class **Same** and  $m(m - 1)q^2/2$  DVs of class **Different**, which means that the ratio between the number of training examples of **Same** and the number of training examples of **Different** is:

$$\frac{q - 1}{q(m - 1)} \approx \frac{1}{m - 1}$$

This indicates that we are in the presence of an imbalanced binary classification problem (which is even more imbalanced if  $m$  is large); however, this is not a problem because, since we typically have many training DVs (see, e.g., Example 2), we can subsample class **Different**, i.e., remove some among its many training examples from the training set.

In sum, the use of the DV-based representation in closed-set AA allows the SAV binary classifier to be trained robustly, thanks to the fact that the existing amount of training information can be devoted to solving a comparatively easier binary classification task rather than a comparatively more difficult 1-of- $m$  classification task. We can thus expect to obtain accurate SAV classification predictions; in Section 5.3.2, we show that the SAV predictions can also be used by a downstream process to solve AA tasks.

### 5.3 Methodology for solving AId tasks with Diff-Vectors

One difference between the standard representation (in which class labels represent authors) and the representation based on DV (in which class labels are either **Same** or **Different**) is that the tasks that can be solved “directly” are AV and AA for the former, and SAV for the latter. That is, by using the standard representation, AV and AA can be solved directly by setting up a classifier that, for a given document, returns a class label in  $\mathcal{A}$  (for AA) or in  $\{A^*, \bar{A}^*\}$  (for AV); SAV is instead to be solved as a derivative, “downstream” task, e.g., by first determining the true authors of documents  $d_i$  and  $d_j$  by means of two calls to an AA classifier, and then checking whether the two returned class labels are the same or not.<sup>2</sup> On the contrary, when using the DV-based representation, SAV can be solved directly; AV and AA are instead to be solved as derivative tasks, using SAV as the building block.

In this section, we first formally define our method for performing SAV using the DV-based representation (Section 5.3.1), and then we describe two alternative solutions for solving both AA and AV that build on top of said SAV method (Section 5.3.2 and 5.3.3).

#### 5.3.1 Solving SAV

Given a training set  $\mathcal{L} = \{(d_1, y_1), \dots, (d_n, y_n)\}$  of documents  $d_i \in \mathcal{D}$  labelled by classes  $y_i \in \mathcal{A} = \{A_1, \dots, A_m\}$  representing authors, we define its pair-based version as:

$$\mathcal{L}_{\mathcal{P}} = \{(d_i, d_j), \text{SD}(y_i, y_j) \mid i, j \in \{1, \dots, n\}, j < i\} \quad (5.5)$$

where  $\text{SD}(y_i, y_j)$  is an indicator function that returns **Same** if  $y_i = y_j$  and **Different** otherwise.

We also assume a feature extractor  $f : \mathcal{D} \rightarrow \mathbb{R}^t$  which maps documents  $d \in \mathcal{D}$  into  $t$ -dimensional vectors  $\mathbf{x}$  of real numbers. We can thus rewrite  $\mathcal{L}$  as  $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$  and redefine  $\mathcal{L}_{\mathcal{P}}$  as:

$$\mathcal{L}_{\mathcal{P}} = \{(\mathbf{x}_{ij}, \text{SD}(y_i, y_j)) \mid i, j \in \{1, \dots, n\}, j < i\} \quad (5.6)$$

where  $\mathbf{x}_{ij} \in \mathbb{R}^t$  is a vector of absolute differences of feature values, i.e.,  $\mathbf{x}_{ij}$  is the vector such that its  $k$ -th component is  $\mathbf{x}_{ij}^k = |\mathbf{x}_i^k - \mathbf{x}_j^k|$ , for all  $1 \leq k \leq t$ ,  $i \neq j$ .

Note that  $|\mathcal{L}| = n$  while  $|\mathcal{L}_{\mathcal{P}}| = n(n-1)/2$ , i.e., the pair-based version  $\mathcal{L}_{\mathcal{P}}$  is  $(n-1)/2$  times larger than its standard counterpart  $\mathcal{L}$ . In practice, the size of  $\mathcal{L}_{\mathcal{P}}$  can be so large as to make the learning process intractable for some batch learners. For example, the 499,500 training DVs of Example 1 would result from a dataset of 10 authors and 100 training documents per author, which is not a terribly large dataset. As shown in Section 5.2.2,  $\mathcal{L}_{\mathcal{P}}$  tends to be imbalanced, with a **Same** / **Different** training example ratio close to  $1/m$  (assuming a training set containing the same number of documents for each author).

In practice, we will be interested in generating and using only a subset  $\mathcal{L}'_{\mathcal{P}} \subset \mathcal{L}_{\mathcal{P}}$ : by including in  $\mathcal{L}'_{\mathcal{P}}$  a small enough number of elements of  $\mathcal{L}_{\mathcal{P}}$ , we can make the training process tractable, and we can avoid the typical negative consequences of imbalance by including in  $\mathcal{L}'_{\mathcal{P}}$  an equal number of examples of **Same** and **Different**. By using a subset  $\mathcal{L}'_{\mathcal{P}}$  with these characteristics, we can then train a binary classifier  $h : \mathbb{R}^t \rightarrow \{\text{Same}, \text{Different}\}$ . We call this classifier DV-Bin, since it is a binary classifier that uses DVs.

Without loss of generality, and for ease of notation, we will henceforth use  $h$  as the function of two arguments  $h : \mathcal{D} \times \mathcal{D} \rightarrow \{\text{Same}, \text{Different}\}$ , thus leaving implicit the phases of (a) mapping documents to

<sup>2</sup>This is possible only for closed-set SAV though, since open-set SAV cannot be recast in terms of AA.

feature vectors, and (b) computing DVs from the absolute differences of feature values. As a result, we can simply write  $h(d_i, d_j)$  to indicate a predicted label in  $\{\text{Same}, \text{Different}\}$ .

### 5.3.2 Solving AA

In this section we describe how SAV can be used to implement AA as downstream tasks.

In order to predict the author of a document  $d$  among the candidates in  $\mathcal{A}$ , and to do so by using a SAV classifier, it makes sense to look at how  $d$  relates to the training documents in terms of the **Same** and **Different** classes. For instance, if for all documents  $d' \in \mathcal{L}$  written by  $A_z$  the pair  $(d, d')$  is assigned by the SAV classifier to class **Same**, and if for all  $d'' \in \mathcal{L}$  written by an author in  $\mathcal{A} \setminus \{A_z\}$  the pair  $(d, d'')$  is assigned to class **Different**, it would be reasonable to predict that  $d$  has been written by  $A_z$ . Unfortunately, this uniformity rarely occurs in practice: in more typical cases the SAV classifier will assign to class **Same**, for example, some pairs  $(d, d')$  where  $d'$  has been written by  $A_z$ , and some pairs  $(d, d'')$  where  $d''$  has been written by an author other than  $A_z$ . Therefore, it is essential to devise a methodology that allows to properly combine such apparently contradictory outcomes.

Moreover, given that we need to build our AA algorithm on top of the output of the SAV classifier, it is in our best interest to squeeze every possible bit of information from this output. As a result, we will be interested in exploiting not just the binary prediction of the SAV classifier, but also its non-binary classification score, representing the degree of certainty of the classifier towards its prediction. To this aim, we assume that our SAV classifier is of the form:

$$h : \mathcal{D} \times \mathcal{D} \rightarrow [0, 1] \tag{5.7}$$

and thus that it returns classification scores that are posterior probabilities. These latter are values  $\Pr(\text{Same}|d_i, d_j)$  that denote the probability that the SAV classifier attributes to the fact that  $d_i$  and  $d_j$  have been written by the same author, and are such that  $\Pr(\text{Different}|d_i, d_j) = 1 - \Pr(\text{Same}|d_i, d_j)$ .

In the next paragraphs, we explore two techniques for building an AA classifier on top of a SAV classifier: the first one is inspired by Lazy Learning methods [14] (Section 5.3.2.1) and the second one is inspired by the well known Stacked Generalisation algorithm [295] (Section 5.3.2.2).

#### 5.3.2.1 LazyAA

We call the first SAV-based AA algorithm of this work LazyAA: it draws inspiration from distance-weighted KNN. Similarly to distance-weighted KNN, the underlying idea of our method is that, given a test document  $d$ , if a training document  $d'$  authored by  $A_z$  is “stylistically similar” to  $d$ , this brings evidence towards the fact that  $d$  is authored by  $A_z$  as well, and this evidence can be quantified exactly by the amount of stylistic similarity.

Unlike distance-weighted KNN, though, instead of having access to a function that computes the similarity between two documents, we here have access to a SAV (soft) classifier that computes the probability that the two documents are in class **Same**. It is thus just natural to compute the stylistic similarity between  $d$  and  $d'$  as  $\Pr(\text{Same}|d, d')$ , i.e., as the probability that the SAV classifier attributes to the fact that  $d$  and  $d'$  have been written by the same author.

Our combination rule thus consists of selecting, for each author  $A_z \in \mathcal{A}$ , the  $k$  training documents written by  $A_z$  that are stylistically most similar to our test document  $d$  (i.e., the ones for which  $\Pr(\text{Same}|d, d')$  is highest), and computing the average value of this stylistic similarity across these  $k$  documents; the au-

thor for which this average stylistic similarity is highest is predicted to be the author of  $d$ . Formally:

$$\begin{aligned} h'(d, \mathcal{L}, k) &= \arg \max_{A_z \in \mathcal{A}} \frac{1}{k} \sum_{d_i \in \text{NC}(k, \mathcal{L}, A_z, d, h)} h(d, d_i) \\ &= \arg \max_{A_z \in \mathcal{A}} \frac{1}{k} \sum_{d_i \in \text{NC}(k, \mathcal{L}, A_z, d, h)} \Pr(\text{Same}|d, d_i) \end{aligned} \quad (5.8)$$

where  $\text{NC}(k, \mathcal{L}, A_z, d, h)$  returns the  $k$  documents from training set  $\mathcal{L}$  that have been written by author  $A_z$  and are closest to  $d$  according to the SAV classifier  $h$ . Note that the  $h'$  functional is parameterised by  $\mathcal{L}$  (and  $k$ ) since, as in all lazy learning methods, there is no proper training phase for  $h'$ , and all the computation is carried out at classification time.

The optimal value for parameter  $k$  can be found via LOO validation on the training set  $\mathcal{L}$ : for each value of  $k$  in the tested range, each training document  $d_i \in \mathcal{L}$  is classified by a classifier  $h'$  trained on  $\mathcal{L} \setminus \{d_i\}$ ;  $k$  is thus set to the value that maximises a given effectiveness measure as computed on the entire set  $\mathcal{L}$ .<sup>3</sup> If we use (vanilla) accuracy (see Appendix A.1) as the effectiveness measure, this process comes down to computing:

$$k^* = \arg \max_k \frac{1}{n} \sum_{(d_i, y_i) \in \mathcal{L}} \mathbf{1}[h'(d_i, \mathcal{L} \setminus \{d_i\}, k) = y_i] \quad (5.9)$$

where  $\mathbf{1}[s]$  is an indicator function returning 1 if statement  $s$  is true and 0 otherwise. This optimisation can be performed very quickly if the posterior probabilities  $\Pr(\text{Same}|d_i, d_j)$  are computed only once for all  $d_i, d_j \in \mathcal{L}$  and stored for fast reuse. Similarly,  $\text{NC}(k, \mathcal{L}, A_z, d, h)$  can be made to return the top  $k$  elements (for different values of  $k$ ) from a fully ranked list that is computed once and reused when necessary.

### 5.3.2.2 StackedAA

We call the second SAV-based AA algorithm of this work StackedAA, since it is inspired by Stacked Generalisation [295]. It consists of an AA (single-label multiclass) classifier  $h'$ , trained by general-purpose learning algorithms, that classifies documents represented by vectors of posterior probabilities  $\Pr(\text{Same}|d, d_k)$ , each of which has been returned by an underlying, previously trained SAV classifier  $h$  (more precisely, a DV-Bin classifier of the type described in Section 5.3.1). More in detail, in order to predict who among the authors in  $\mathcal{A}$  has written document  $d$ , we represent  $d$  via a vector:

$$\begin{aligned} \phi(d) &= (h(d, d_1), \dots, h(d, d_n)) \\ &= (\Pr(\text{Same}|d, d_1), \dots, \Pr(\text{Same}|d, d_n)) \end{aligned} \quad (5.10)$$

of  $n$  posterior probabilities, one for each training example in  $\mathcal{L}$ . The  $k$ -th value in this vector is the value  $h(d, d_k) = \Pr(\text{Same}|d, d_k)$ , where  $d_k$  is the  $k$ -th training example. In other words, in order to classify  $d$  we first need to perform  $|\mathcal{L}|$  SAV classifications, where the  $k$ -th such classification attempts to predict whether the test document  $d$  was written by the same author who also wrote training document  $d_k$ .

At training time, we train the AA classifier  $h'$  by using all the training examples in  $\mathcal{L}$  represented in the style of Equation 5.10. In other words, by applying the mapping  $\phi : \mathbb{R}^t \rightarrow [0, 1]^n$  to the training documents themselves we define a new “view”  $\mathcal{L}_h = \{(\phi(d_i), y_i)\}_{i=1}^n$  of the training set  $\mathcal{L}$ , in which the training documents are not represented via vectors of  $t$  stylistic features, but via vectors of  $|\mathcal{L}|$  posterior

---

<sup>3</sup>One might wonder why we go for LOO, a traditionally expensive way of optimising parameters, rather than the cheaper  $t$ -fold cross-validation ( $t$ -CV). The reason is that, in our case, LOO is no more expensive than  $t$ -CV, because we are in a lazy learning context. In fact, in traditional eager learning contexts, LOO requires  $|\mathcal{L}|$  classifier retrainings, while  $t$ -CV requires only  $t \ll |\mathcal{L}|$  classifier retrainings; however, in lazy learning contexts there are no retrainings because classifiers are not “trained”, since all inductive inference is carried out at classification time.

probabilities, with  $\phi(x) \in [0, 1]^n$ . The training set  $\mathcal{L}_h$  can directly be used to train a general-purpose classifier  $h' : [0, 1]^n \rightarrow \mathcal{A}$  in the feature space of posterior probabilities.<sup>4</sup> Of course, in order to generate  $\mathcal{L}_h = \{(\phi(d_i), y_i)\}_{i=1}^n$  we first need to train a SAV classifier  $h$  via the DV-Bin method of Section 5.3.1. In the experiments of Section 5.5 we will concentrate on instantiations of  $h'$  that are generated by the same learning method used to generate  $h$ . At classification time, a given test document  $d$  is classified by first computing  $\phi(d)$  (this requires invoking  $n$  times classifier  $h$ ) and then invoking classifier  $h'(\phi(d))$ .

StackedAA differs from Stacked Generalisation in one important aspect: in Stacked Generalisation the metaclassifier and the base classifiers are homogeneous, i.e., they all use the same set of classes, while in StackedAA the metaclassifier and the base classifiers are heterogeneous, i.e., they use different sets of classes. Indeed, the base classifiers use the classes in  $\{\text{Same, Different}\}$ , since they are binary SAV classifiers, while the metaclassifier use the classes in  $\mathcal{A} = \{A_1, \dots, A_n\}$ , since it is a single-label multiclass AA classifier.

There are several important aspects in which the two methodologies, StackedAA and LazyAA, differ:

- in StackedAA, evidence is provided by all training examples, and not just by the  $k$  examples most similar to the test example, as is instead the case in LazyAA;
- in StackedAA, the combination rule (i.e., the rule that assembles the evidence provided by the training examples into a final decision) is learnt by a metaclassifier, i.e., it is not static, as is instead the case in LazyAA;
- in StackedAA, learning is performed offline (since the metaclassifier is trained before the testing phase begins), while in LazyAA all inductive inference is carried out at classification time.

### 5.3.3 Solving AV

It is fairly straightforward to take the algorithms described in Sections 5.3.2.1 and 5.3.2.2 and generate versions that solve AV instead of AA. We call them LazyAV and StackedAV, respectively.

The only difference between LazyAV and LazyAA, and between StackedAV and StackedAA, is that in the AV versions the codeframe used is binary, i.e., it is  $\mathcal{A} = \{A^*, \bar{A}^*\}$ ; in particular, this means that for StackedAV the metaclassifier  $h'$  is a binary classifier instead of a multiclass classifier. Everything else is unmodified.

However, in preliminary experiments, both LazyAV and StackedAV proved substantially inferior to LazyAA and StackedAA versions, respectively, in which the document  $d$  is attributed to  $A^*$  if the AA algorithm does so, and is attributed  $d$  to  $\bar{A}^*$  if the AA algorithm attributes it to an author  $A_z$  different from  $A^*$ . Concerning why LazyAV underperforms LazyAA, it might be due to the fact that there is an *a priori* high probability that the  $k$  nearest neighbours in  $\bar{A}^*$  are, on average, closer to  $d$  than the  $k$  nearest neighbours in  $A^*$ , since  $\bar{A}^*$  is a very large pool to choose from (this does not happen in AA, where, assuming an equal number of training documents per author, all pools are equally large); this can give undue advantage to  $\bar{A}^*$  over  $A^*$ , and thus generate a large quantity of false negatives. Concerning why StackedAV underperforms StackedAA, it might be due to the fact that the metaclassifier of StackedAV does not put the available class information to the best use, conflating all labels different from  $A^*$  into a single label  $\bar{A}^*$  that ends up being poorly characterised from the semantic point of view.

Therefore, in the rest of this work the algorithms we use for solving AV via the DV-based representation is the versions of LazyAA and StackedAA described at the beginning of the previous paragraph. A

---

<sup>4</sup>Note that, if the learning algorithm is a linear model, then it takes the form of  $h'(d) = \sum_{i=1}^n \alpha_i h(d, d_i)$ , in which  $\{\alpha_i\}_{i=1}^n$  are the parameters to be learned, and the set of functions  $\{h(\cdot, d_i)\}_{i=1}^n$  plays the role of a set of basis functions centred at the training points.

consequence of this is that any AA experiment that involves the use of either LazyAA and StackedAA and a codeframe  $\mathcal{A} = \{A_1, \dots, A_m\}$ , is also *de facto* a set of  $m$  different AV experiments. In other words, we do need to run separate AA and AV experiments, and thus we evaluate the AA experiments that we describe in Section 5.5.2 both in terms of AA and AV.

## 5.4 Experimental setting

In order to test whether a representation based on DVs is advantageous with respect to a representation based on standard vectors, we compare these two different design choices in experiments that we run on four publicly available datasets (one, assembled by us, is made available here for the first time) and for all three authorship analysis tasks (AA, AV, SAV).

In the following paragraphs, we describe the datasets we use (Section 5.4.1), and the learning algorithms and feature sets we employ (Section 5.4.2).

### 5.4.1 Datasets

We run experiments on four datasets consisting of textual documents annotated by author. These datasets are representative of different textual genres, lengths, and styles (all in the English language), and they are publicly available. The four datasets are:

- IMDB62. This dataset contains movie reviews, see Appendix B.5 for the full description of this dataset. In order to divide the 62,000 documents into a training set and a test set, we perform a stratified split, resulting in 700 training documents and 300 test documents for each author. We use these texts as examples of a “moderately formal” type of communication, since the reviews are not as short as, for example, online messages, and, despite some occasional slang, are written in a clear and correct (although often informal) manner.
- PAN11. This dataset is based on the Enron email corpus, and was developed for the PAN-2011 international Authorship Identification competition [20]; see Appendix B.6 for the full description of this dataset. In these experiments, we use the LARGE problem setting, with its training and test sets. The emails are often extremely short, and show many characteristics of online communication; in order to avoid texts which are excessively short (and thus too difficult to attribute), we remove emails consisting of fewer than 15 words. We thus end up with 7,111 training documents and 1,157 test documents, altogether accounting for 70 different authors.
- VICTORIAN. This dataset is made of segments of books from American or British 18th-19th century novelists; see Appendix B.7 for the full description of this dataset. In order to divide it into a training set and a test set, we perform a stratified split, including 70% of each author’s texts in the training set and the remaining 30% in the test set. We use these documents as examples of literary production characterised by a sophisticated style.
- ARXIV. This dataset was created by us for this project. It consists of abstracts of single-author papers from ARXIV; see Appendix B.8 for the full description of this dataset. In order to divide the corpus into a training set and a test set, we perform a stratified split, with the production of each author being split into a training set (70% of the abstracts) and a test set (30%). We use these abstracts as examples of “scientific communication”, characterised by a precise and compact style, with an abundance of technical terminology.

Table 5.1 summarizes the main characteristics of these four datasets.

Table 5.1: Main characteristics of the datasets used in this work. The table shows: the name of the dataset (**Name**), the article the dataset was first proposed in (**Proposed in**), the type of documents it contains (**Types of docs**), the number of authors (**# authors**), the minimum/maximum number of docs per author (**Min # docs / Max # docs**), the total number of documents in the dataset (**Total # docs**) and the average length of the documents in the dataset (**Avg length**).

Name	Proposed in	Type of docs	# authors	Min # docs	Max # docs	Total # docs	Avg length
IMDB62	[262]	film reviews	62	1,000	1,000	62,000	349.0
PAN11	[20]	emails	70	1	561	8,268	69.2
VICTORIAN	[114]	[segments of] novels	45	183	6,914	53,678	1,000.0
ARXIV	[this work]	abstracts of papers	100	10	34	1,469	129.3

## 5.4.2 Learners and features

In this study, we chose LR as our learning method, because it is known to deliver very good accuracy in many text-related applications, and because the outputs are well-calibrated probabilities (see Section 2.3). This is a very important advantage, since the methods we describe in Sections 5.3.2.1 and 5.3.2.2 do rely on posterior probabilities, and obviously benefit from the fact that these posteriors are of high quality.

We optimise the hyperparameter  $C$  (the inverse of the regularisation strength) in the log-space  $\{10^i\}_{i=0}^{i=4}$ , and select the value of  $C$  that minimizes the multinomial loss in a stratified 5-CV. We use the `LogisticRegressionCV` implementation from the `scikit-learn` package (see Appendix D)

In order to generate the **Same** and **Different** training pairs, we adopt the following policy. Given a training set  $\mathcal{L}$ , we first compute the number of **Same** pairs that can be generated. If there are fewer than 50,000 **Same** pairs, we generate them all; otherwise, we draw (uniformly at random) 50,000 **Same** pairs from the total. We then draw (again, uniformly at random) as many **Different** pairs as the **Same** pairs we have generated. This is in order to guarantee a balanced training set, since there are usually many more potential **Different** pairs than **Same** ones.

As for the choice of features, we stick to ones well-known and broadly adopted in the field of AAn, since they are features of a frequentistic nature that can be extracted automatically and that are believed to convey stylistic information (for a discussion regarding these features, and more generally regarding feature design, see Section 2.1). Note that other sets of features could have been equally plausible; however, this is not an important concern for our work, since it is completely agnostic with respect to the specific features that should be used. Specifically, the features we use can be naturally subdivided into two sets. Set 1 is composed of:

- **Function words:** we use the list of English function words provided by NLTK (see Appendix D);
- **Word lengths:** We consider each word length instantiated in the training set as a feature;
- **Sentence lengths:** We consider each sentence length instantiated in the training set as a feature;
- **Punctuation symbols:** We consider each punctuation symbol that occurs in the training set as a feature.

Set 2 is composed of:

- **POS  $n$ -grams:** we extract POS tags from our texts by using the `spaCy` library (see Appendix D), and we consider each POS  $n$ -gram (for  $n \in [3, 4]$ ) that occurs in the training set as a potential feature (see below);
- **Word uni-grams:** we consider each word that occurs in the training set as a potential feature;
- **Character  $n$ -grams:** we consider each character  $n$ -gram (for  $n \in [2, 5]$ ) that occurs in the training set as a potential feature.

The features in Set 1 are relatively few (typically:  $O(10^2)$ ), and are dense, so that all of them can be expected to occur to some degree in most texts. Given one of these features and given a document, we take its relative frequency in the document as the value of the feature in the document. We also apply standardisation to the columns that these features generate in the document-by-feature matrix.<sup>5</sup>

The features in Set 2 are plenty (typically:  $O(10^4)$  or  $O(10^5)$ ). In order to deal with the fact that they may be *too* many, we apply filter-style feature selection to them, using the Chi-square test (see Appendix A.2) as the term scoring function and retaining the 50,000 highest-scoring features. As the feature weighting function, rather than using plain relative frequency, we use TfIdf (see Appendix A.3). The features in Set 2 are sparse: a large number of them will not occur in a given document. We do not apply any standardisation to the features in Set 2, since this would turn them into dense features, and this would be detrimental to efficiency.

### 5.4.3 Experimental protocol

In this analysis, we conduct two sets of experiments using DVs. The first set, referred to as “intrinsic” evaluation, involves a series of SAV experiments (both open- and closed-set) to assess the classifier’s performance on SAV tasks, which can be solved directly. The second set, referred to as “extrinsic” evaluation, comprises a series of closed-set AA experiments; in this set, we evaluate the two methods that utilize a SAV classifier as a foundational component that we described in Section 5.3.2.

For both evaluations, we consider a set of authors  $\mathcal{A} = \{A_1, \dots, A_m\}$ , with  $m > 2$ , each one being the author of  $q$  training documents, and a dataset that contains a test set  $\mathcal{U}$ .

- **Intrinsic evaluation.** We test our systems on randomly drawn samples of test document pairs from  $\mathcal{U}$ . The reason why we do not test on all possible pairs is a practical one, since the number  $|\mathcal{U}|(|\mathcal{U}|-1)/2$  of all possible pairs is too high for most datasets. We randomly draw balanced subsets of 1,000 test pairs (500 positive and 500 negative) for each experiment.

We investigate the impact on performance of the number  $m$  of authors and the number  $q$  of training documents per author. Specifically, for the IMDB62, PAN11, and VICTORIAN datasets we run experiments varying the number  $m$  of authors in the set  $\{5, 10, 15, 20, 25\}$ , and the number  $q$  of documents per author in the set  $\{10, 20, 30, 40, 50\}$ . Samplings are incremental; e.g., when moving from  $q = 20$  to  $q = 30$ , we add 10 new documents per author to the previous 20. Regarding the test set, for each choice of  $m$  we draw 1,000 random pairs consisting of texts written by some among the  $m$  authors present in the training set (which we use to evaluate the closed-set SAV setting), and 1,000 random pairs consisting of texts written by  $m$  authors other than the  $m$  authors present in the training set (which we use to evaluate the open-set SAV setting).<sup>6</sup>

In order to compensate for the random effect introduced by sampling (authors, documents, and test pairs), we report results obtained by averaging across 10 runs for each combination  $\langle \text{dataset}, m, q \rangle$ ; we use the same random samples for all the methods we compare. The only exception is the ARXIV dataset, which, due to its limited size, does not allow this extraction of multiple samples; hence, for this dataset we simply report experiments across 10 random train/test splits of the entire dataset.

---

<sup>5</sup>Standardisation (a.k.a., z-scoring) is a normalisation process consisting of centering and scaling a random variable so as to force its distribution to have 0-mean and 1-variance, i.e., the z-score of a raw variable  $x$  is defined as  $z = \frac{x-\mu}{\sigma}$  where  $\mu$  and  $\sigma$  are the (sample) mean and (sample) standard deviation of  $x$  as estimated in the training set. For the benefits in accuracy deriving from standardising dense features, see Moreo et al. [206].

<sup>6</sup>As detailed in Section 1.1, in open-set SAV one normally assumes that the authors of the two unlabelled documents are not necessarily among the authors represented in the training set; in these experiments we consider the more difficult setting in which the authors of the two unlabelled documents are strictly not among the authors represented in the training set.

We evaluate the performance in terms of Acc (see Appendix A.1), which is a valid evaluation measure when the test set is balanced across the classes, such as the present one.

- **Extrinsic evaluation.** The method is asked to attribute each test document in  $\mathcal{U}$  to one of the authors in  $\mathcal{A}$ , in a single-label multiclass fashion.

We investigate the impact on AA accuracy of the number  $m$  of authors and the number  $q$  of documents per author. We let  $m$  take values in the set  $\{5, 10, 15, 20, 25\}$  as before, and we let  $q$  take values in the set  $\{5, 10, \dots, 45, 50\}$ .<sup>7</sup> As previously, the experiments are different for the ARXIV dataset, in which the above fine-grained exploration is not possible. For both LazyAA and StackedAA we use the DV-Bin classifier described in Section 5.3.1 as the underlying SAV mechanism.

We use  $F_1$  (see Appendix A.1) as evaluation measure, since not all our datasets are balanced, and Acc is a notoriously bad measure for working with imbalanced datasets, unlike  $F_1$ . For all datasets, we report the values of  $F_1^M$ ; in the case of the ARXIV dataset, we also report micro-averaged  $F_1^\mu$ , since this is the only imbalanced dataset we employ (and since micro-averages would coincide with macro-averages in the balanced datasets IMDB62, PAN11, and VICTORIAN). All results are reported as averages across 10 runs that use different random seeds.

## 5.5 Results

In the following paragraphs, we show the results of our analysis regarding the employment of DV-based representation for solving the three major AId tasks; in particular, in Section 5.5.1 we discuss our “intrinsic” evaluation of DV (regarding SAV, which is naturally solved by DVs), while in Section 5.5.2 we discuss an “extrinsic” evaluation of DVs (regarding the downstream tasks of AV and AA).

Moreover, in Section 5.5.3 we consider some relevant question regarding the efficiency and the computational time cost when employing DVs. In Section 5.5.4, we comment on the applicability of DVs for an AV setting where (unlike the previous investigation) the author labels for the negative class are unknown. Finally, in Section 5.5.5 we present the results of a brief experimentation with a learner different from LR (we adopt SVM).

### 5.5.1 Intrinsic evaluation of Diff-Vectors

We here show the results for our intrinsic evaluation of DV, where we employ DVs to tackle the SAV task, which can be solved directly by DVs. We perform experiments in both the closed-set SAV setting (Section 5.5.1.1), where the authors in the test set are the same as in the training set, and in the open-set SAV setting (Section 5.5.1.2), where the authors in the test set are strictly not appearing in the training set.

#### 5.5.1.1 Experiments for closed-set SAV

In the closed-set scenario, the authors in the test set  $\mathcal{U}$  are the same as in the training set. We here explore two variants of our method:

- **DV-Bin:** the binary classifier discussed in Section 5.3.1.

---

<sup>7</sup>In this case, we explore a finer-grain grid for  $q$  than in our previously discussed SAV experiments, because in this evaluation we are not considering IM as a competitor (see the following Section 5.5), and thus these experiments are considerably faster to run.

Table 5.2: Intrinsic evaluation of DV: results on closed-set SAV using Acc as the evaluation measure on dataset ARXIV. **Boldface** indicates the best method. Symbols \* and \*\* denote the method (if any) whose score is not statistically significantly different from the best one at  $\alpha = 0.05$  (\*) or at  $\alpha = 0.001$  (\*\*) according to a paired sample two-tailed t-test. No symbols \* and \*\* appear in this particular table since all differences are statistically significant.

	mean	std	ttest
DV-Bin	.756	0.017	
DV-2xAA	<b>.803</b>	0.025	
STD-CosDist	.629	0.022	
STD-2xAA	.646	0.014	

- **DV-2xAA**: a method that solves SAV by building on top of the LazyAA method discussed in Section 5.3.2.1. In other words, this method first predicts, for both unlabelled documents, who the author of the document is, and then checks if the two predicted authors are the same author.

We consider the following baseline systems:

- **STD-CosDist**: a binary classifier trained to predict whether the pair belongs to Same or Different, where a pair of documents is represented by a vector of one feature only. The value of this feature is obtained by calculating the distance between the two documents, each represented by a standard vector; the distance function is the cosine distance.<sup>8</sup> The training set is transformed into pairs following the same policy as in DV-Bin (see Section 5.3.1). The classifier thus learns the distance threshold that best separates the Same pairs from the Different pairs.
- **STD-2xAA**: a single-label multiclass classifier that operates on standard vector representations and that, as in DV-2xAA, solves SAV by performing closed-set AA for both documents and then checking if the two predicted authors are the same.

Both baselines are equipped with the same learner as our method: LR optimised by running the optimisation process for hyperparameter  $C$  described in Section 5.4.2.

Figure 5.1 reports the results for this experimentation, displayed in terms of Acc (on the  $y$  axis) as a function of the number of training documents per author (on the  $x$  axis), in datasets IMDB62, PAN11, and VICTORIAN (each corresponding to a different column), at varying number of authors (each corresponding to a different row).<sup>9</sup> Each coloured dot represents an average result across 10 experiments, while the colour band frontiers indicate  $\pm$  one standard deviation from the mean. Table 5.2 reports the results for the ARXIV dataset.

These results clearly indicate that the DV-based variants perform well: between the DV-2xAA and the STD-2xAA (the two methods achieving SAV by running AA on both documents), the DV-based method is always better or much better than the standard vector-based method, and the same happens between the DV-Bin and the STD-CosDist methods (the two non-AA-based methods). The top-performing method is unquestionably DV-2xAA, which always outperforms (often by a very large margin) all others, for all numbers  $m$  of authors and for all numbers  $q$  of training examples per author. As for the reason why DV-2xAA outperforms DV-Bin, we conjecture that this may happen because the LazyAA method uses only evidence conveyed by few relevant training documents (the  $k$  documents most similar to the test document, for both test documents), thus filtering out other less relevant documents. All algorithms obviously improve their performance as the number of documents per author increases, with the sole

<sup>8</sup>We have also run experiments using the L1 or L2 distances in place of the cosine distance; we omit to report their results since cosine proved the best-performing one.

<sup>9</sup>Note that the combination  $\langle \text{PAN11}, 25, 50 \rangle$  is not feasible, given that in PAN11 there are fewer than 50 authors (25 for the closed-set setting and 25 for the open-set setting) with at least 50 training documents each, and thus it is not reported here.

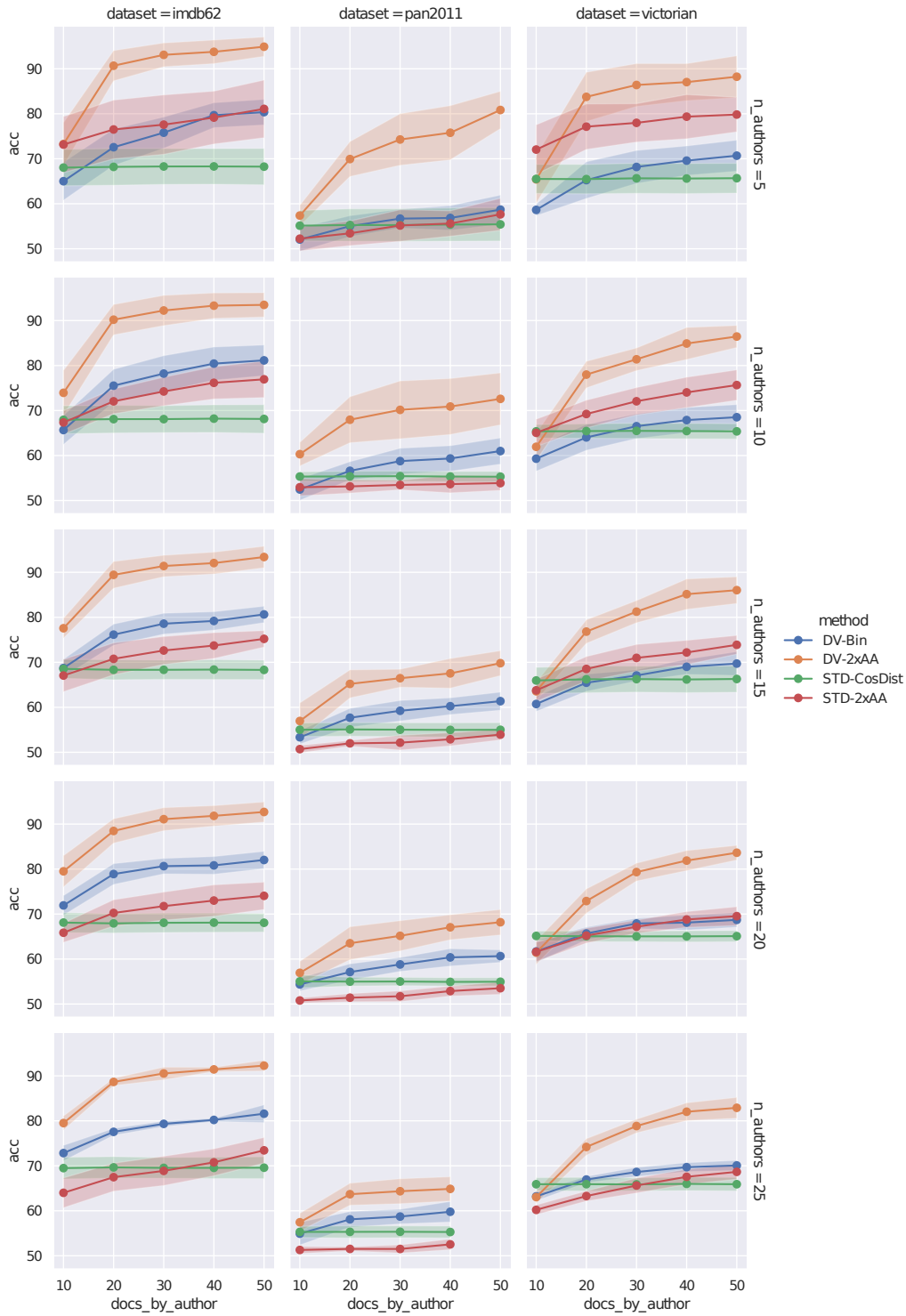


Figure 5.1: Intrinsic evaluation of DV: results on closed-set SAV, using Acc (on the  $y$  axis) as the evaluation measure on datasets IMDB62, PAN11, and VICTORIAN.

exception of STD-CosDist. This latter fact might indicate that the optimal distance threshold that STD-CosDist finds is fairly stable, and is well estimated even by using few training data. However, it seems clear from these results that distances alone do not carry as much information as DVs do.

Figure 5.2 shows the distribution of  $\Pr(\text{Same}|d', d'')$  values for Same and Different pairs that STD-CosDist and DV-Bin compute. For this experiment we have set  $m = 20$  and  $q = 50$  for all datasets

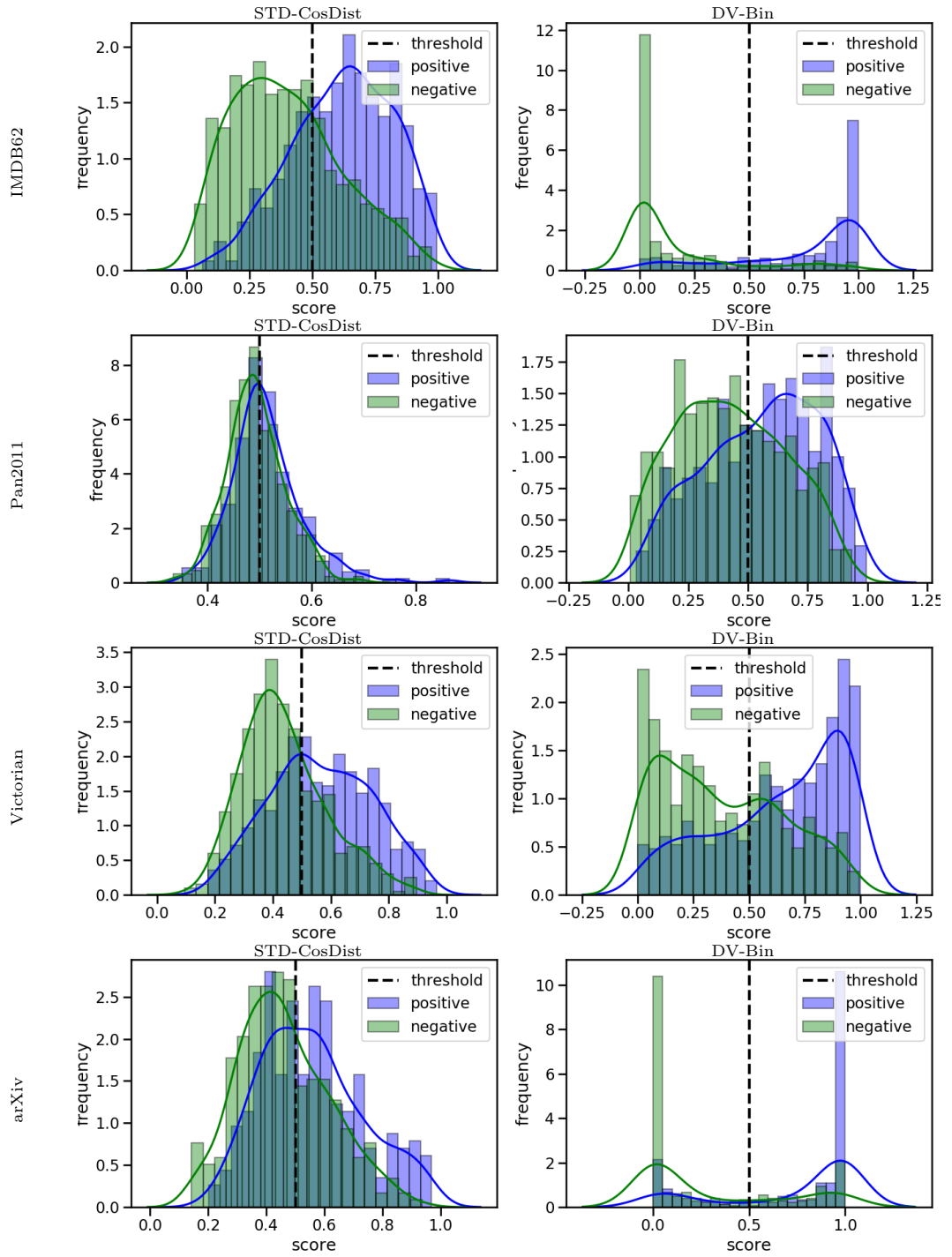


Figure 5.2: Distribution of the decision scores for Same and Different pairs as computed by STD-DistCos (first column) and DV-Bin (second column).

except for ARXIV, where we have set  $m = 50$  and used all the documents written by the 50 authors.<sup>10</sup> The STD-CosDist method manages to separate the posteriors of the Same and Different pairs to some extent in the IMDB62 and VICTORIAN datasets, but it fails to separate them well in PAN11 and ARXIV. Interestingly enough, the posteriors generated by STD-CosDist are close to being normally distributed, both for the Same pairs and for the Different pairs. However, things are very different for the DV-Bin method, which tends to generate much more polarised scores (i.e., separate the positives from the negatives much better), placing most of the density mass around 0 for Different pairs and around 1 for Same pairs, which is indicative of a very good performance. Still, the score distribution generated for VICTORIAN and (especially) for PAN11 reveal that the DV-Bin method still has room for improvement.

### 5.5.1.2 Experiments for open-set SAV

In the open-set SAV experiments, there is no intersection between the set of  $m$  authors that compose the test set and the set of  $m$  authors observed during training. This aspect automatically rules out any attempt to perform SAV via AA (i.e., the DV-2xAA method); for this reason, in this setting the only DV-based method we test is DV-Bin (the binary classifier discussed in Section 5.3.1). The baseline systems we consider are:

- **STD-CosDist**: the same distance-based method that we have used in the closed-set SAV experiments (see Section 5.5.1.1). In this case the method is constrained to learn the optimal threshold from authors different from those in the test set.
- **IM**: the method developed by Koppel and Winter [176], discussed in Section 2.2. We use our own implementation of the “bloggers” variant, which had proved superior to others in their experiments of [176]; it uses documents from the same domain (blogs in the original experiments, documents from the training set in our case) as the impostors candidates. We use cosine as distance function, since in our experiments we have found it to consistently deliver better results than the “minmax” criterion (the similarity function of choice in Koppel and Winter [176]). We set parameter  $I$  (the number of impostor candidates) to 50 instead of 250 (which was found to work well by Koppel and Winter [176]), since our training sets are much smaller than those they consider, the rest of the parameter values we use, following Koppel and Winter [176], are  $i = 10$  (number of impostors) and  $k = 100$  (number of feature subsets). Also following the original paper, we optimise parameter  $\sigma^*$  (the decision threshold) on a validation set. Note that we do not use this baseline method in the closed-set SAV experiments, since in that case the impostors cannot be created. The reason is that the method would likely consider the training documents of one of the test authors as candidate impostors (since these training documents are expected to be more similar to the test document). As a result, the test author could wrongly be identified as an impostor of themselves.

Figure 5.3 displays the results for IMDB62, PAN11, and VICTORIAN,<sup>11</sup> while Table 5.3 reports the results obtained for the ARXIV dataset.

No clear winner emerges in the light of these results. DV-Bin seems to perform best on IMDB62, especially when the number of authors increases, while STD-CosDist seems to perform slightly better on VICTORIAN; all methods perform comparably on PAN11 and ARXIV. Somehow surprisingly, the IM method seems to not take advantage of the increased number of documents per author, likely because the number of actual impostors ( $i = 10$ ) is set in advance, and thus the method is indifferent to variations

<sup>10</sup>Note that the 2xAA variants do not compute a single posterior probability and are thus not amenable to a similar analysis.

<sup>11</sup>The combination  $\langle \text{PAN11}, 25, 50 \rangle$  is not reported here, given the reasons in Footnote 9 in Section 5.5.1.1.

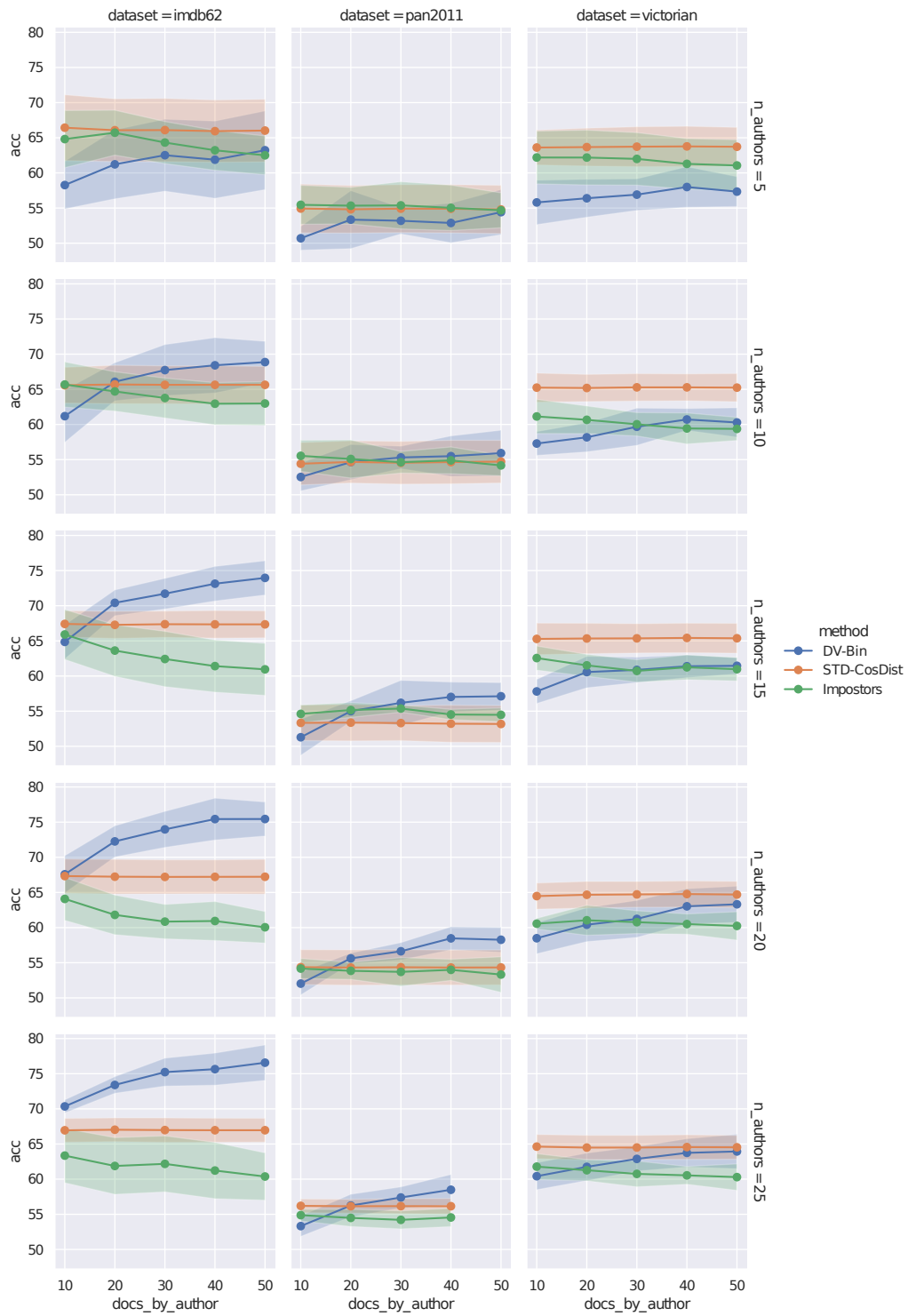


Figure 5.3: Intrinsic evaluation of DV: results on open-set SAV, using Acc (on the  $y$  axis) as the evaluation measure on datasets IMDB62, PAN11, and VICTORIAN.

in  $q$ . DV-Bin tends to perform poorly when the number of documents per author is very small (i.e., 10); this is likely due to the limited number of Same pairs that can be generated from just 10 elements. Concerning STD-CosDist, it proves a fairly stable method, as in the closed-set scenario. PAN11 proves the most challenging dataset here, with all methods performing only marginally better than a random classifier (which would obtain an expected accuracy of 0.50). Regarding the ARXIV dataset, DV-Bin performs best on average, but the t-test reveals that this superiority is not statistically significant.

Table 5.3: Intrinsic evaluation of DV: results on open-set SAV using Acc as the evaluation measure on dataset arXiv. The notational conventions are the same as in Table 5.2.

	mean	std	ttest
DV-Bin	<b>.663</b>	0.020	
STD-CosDist	.661	0.019	**
IM	.642	0.025	**

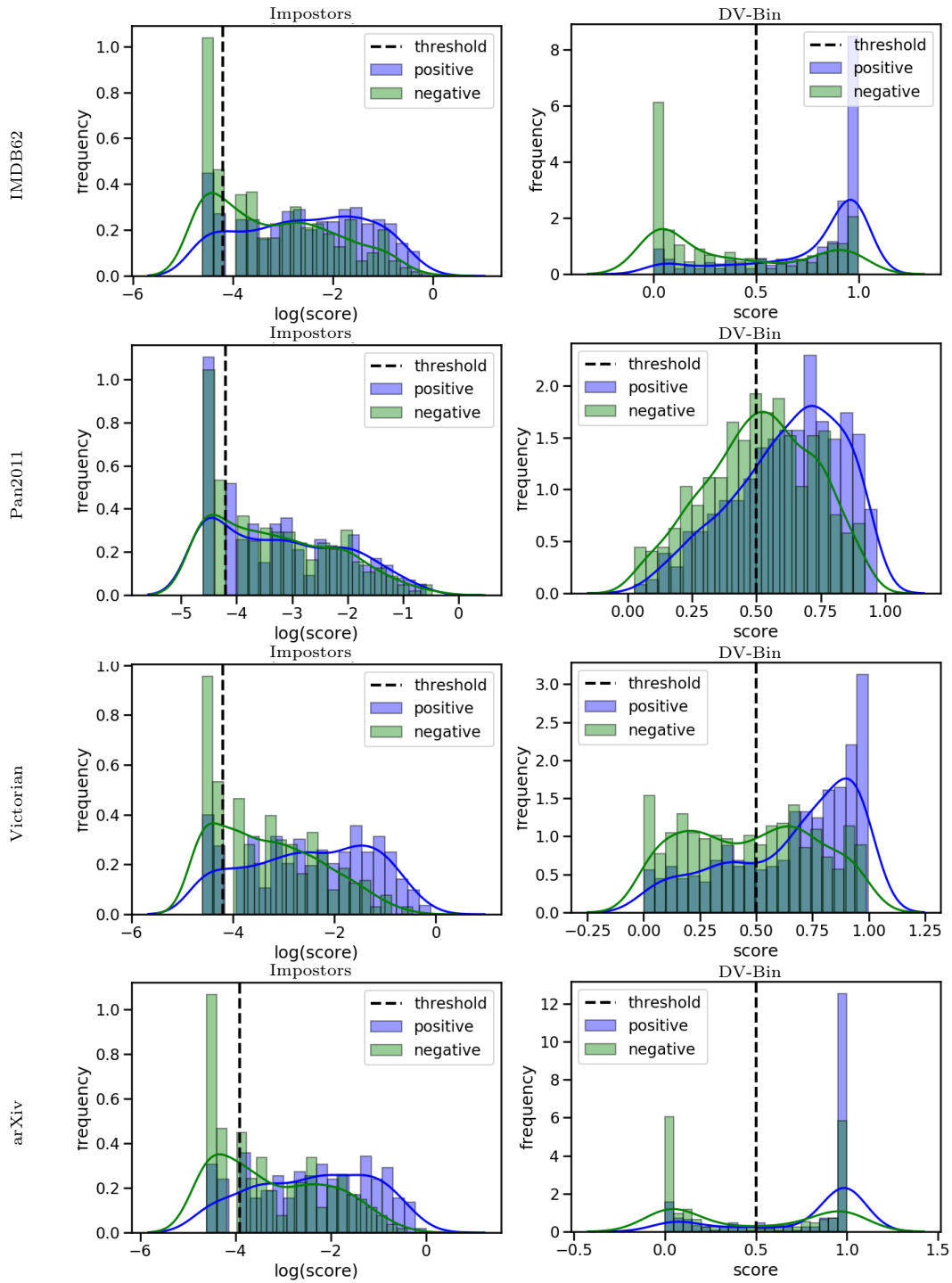


Figure 5.4: Distribution of decision scores for Same and Different pairs as computed by the Impostors method (1st column; note the log scale) and by the DV-Bin method (2nd column).

Summing up, there is no strong enough empirical evidence to claim that the DV-Bin method outperforms IM in open-set SAV. However, there are some technical reasons why one should prefer the DV-Bin method to the IM method. The first concerns its efficiency: IM is a lazy method, meaning that it has no offline training phase (all inductive inference is carried out in the classification phase), and the workload that a single test pair entails is significant, since it involves computing the similarity between the test document and each training document, and computing  $k$  rounds of bagging for each impostor and for each element in the pair. Conversely, once trained, classifying an unlabelled pair using DV-Bin comes down to computing a simple linear combination of feature differences.<sup>12</sup> The second reason concerns its applicability: by definition, the IM method cannot be used, as observed above, in closed-set SAV and, more generally, in SAV settings in which documents written by any of the authors of the test pair are observed in training.

Figure 5.4 shows the distribution of the posteriors  $\Pr(\text{Same}|d', d'')$  for **Same** and **Different** pairs that IM and DV-Bin compute. As for closed-set SAV, we set  $m = 20$  and  $q = 50$  for all datasets except ARXIV, for which we instead set  $m = 50$  and keep all documents per author. Recall that, in our open-set setting,  $m$  specifies both the number of authors involved in the training set and the number of authors involved in test (e.g., in the case of ARXIV, we are using the entire dataset since there are 100 distinct authors). For ease of visualisation, we report the score values according to a logarithmic scale.

The IM method produces decision scores which tend to be very close to 0. The dashed vertical line indicates the decision threshold found optimal in the validation phase; this threshold is  $\sigma^* = 0.005$  in all cases but in ARXIV, where  $\sigma^* = 0.01$ . Note that this threshold succeeds in placing most of the negative scores below it, but still misclassifies many positives. Particularly, in PAN11 and ARXIV it fails to push many of the positive scores beyond the decision threshold. The DV-Bin method instead succeeds at polarising the decision scores of **Same** and **Different** pairs in IMDB62 and ARXIV, although it fails to allocate most of the negative mass below the 0.5 threshold in VICTORIAN and, to a greater extent, in PAN11. Overall, as clear from a simple visual inspection, the DV-Bin method is better than IM at correctly separating the scores of the **Same** pairs from those of the **Different** pairs on each of our four datasets.

## 5.5.2 Extrinsic evaluation of Diff-Vectors

Our extrinsic evaluation of DVs consists of closed-set AA experiments, which we show in Section 5.5.2.1. We do not run specific experiments for DVs applied to AV, since, as discussed in Section 5.3.3, each of our AA experiments is also a set of  $m$  AV experiments, and can be evaluated as such, which we do in Section 5.5.2.2.<sup>13</sup>

### 5.5.2.1 Experiments for AA

At the core of our AA methods, there is a SAV classifier that operates on pairs of documents. Given a test document  $d$ , its attribution is performed by applying a combination rule to the posterior probabilities generated for pairs of documents consisting of the test document  $d$  and a training document  $d'$ . In particular, we explore two methods:

<sup>12</sup>Of course, it is fair to mention that the IM method incurs no cost for training, but this only applies if the value of the parameter  $\sigma^*$  is hard-wired. In practice, the optimal  $\sigma^*$  has to be estimated in a validation phase, which amounts to using a training set to perform repeated rounds of tests, which, as indicated above, require a considerable computational effort.

<sup>13</sup>Note that here we assume to know, at training time, the paternity (i.e., the labels) of documents written by authors other than  $A^*$ . Alternatively, AV can be formulated as a problem in which  $m = 2$  and  $\mathcal{A} = \{A^*, \bar{A}^*\}$ , in which class  $\bar{A}^*$  collectively represents the production of authors other than  $A^*$ . This special case will be discussed more in detail in Section 5.5.4.

Table 5.4: Extrinsic evaluation of DV: results on closed-set AA using  $F_1$  as the evaluation measure on dataset ARXIV. The notational conventions are the same as in Table 5.2. No symbols \* and \*\* appear in this particular table since all differences are statistically significant.

	$F_1^M$			$F_1^\mu$		
	mean	std	ttest	mean	std	ttest
LazyAA	<b>0.682</b>	0.024		<b>0.676</b>	0.019	
StackedAA	0.639	0.028		0.639	0.024	
STD-AA	0.374	0.018		0.405	0.021	
STD-Bin	0.235	0.014		—	—	

- **LazyAA**: the lazy combination rule inspired by KNN discussed in Section 5.3.2.1.
- **StackedAA**: the linear combination rule inspired by Stacked Generalisation discussed in Section 5.3.2.2.

As baseline, we consider STD-AA, a single-label multiclass classifier trained to distinguish among the  $m$  classes from the observation of standard vectors of features. Given a test document, the classifier returns the author which obtains the maximum posterior probability.

Figure 5.5 displays the experimental results for the IMDB62, PAN11 and VICTORIAN datasets (for the moment, let us disregard the curves for STD-Bin, on which we will comment later).<sup>14</sup> The first three rows of Table 5.4 report the results obtained on the ARXIV dataset.

These results show the drastic superiority of DVs over standard vectors for AA. Only for  $q = 5$ , and infrequently for  $q = 10$ , does STD achieve (marginally) better results than the DV-based variants. In this case, the reason might be that low values of  $q$  result in fewer Same pairs (e.g., for  $q = 5$  there are only 10 unordered pairs), which might lead to suboptimal accuracy for the underlying SAV methods. Regarding our variants, the KNN-inspired combination rule consistently outperforms the linear one in IMDB62 and ARXIV, and is slightly better or comparable in the rest of the cases. All methods understandably benefit from the increase in  $q$ , but DVs seem to do so at a much greater rate; indeed, the increase in the number of training examples is quadratic in  $q$  for the DV-based variants, while it is linear in  $q$  for STD.

### 5.5.2.2 Results for AV

Concerning the AV task, note that  $F_1^M$  is also a suitable measure for evaluating AV. In fact,  $F_1$  as measured on a specific author  $A^*$  is a suitable measure for evaluating AV once  $A^*$  is considered the candidate author, and  $F_1^M$  is a suitable measure for computing the average performance for all possible choices of  $A^*$ . As a consequence, the results reported in Figure 5.5 and Table 5.4 also count as an evaluation of the reported methods for the AV task.

For AV, we add another baseline, STD-Bin: it consists of a binary classifier trained to distinguish between  $A^*$  and  $\bar{A}^*$  from the observation of standard vectors of features. It is fair to add this baseline since it would be just natural to solve AV with a binary classifier, instead of using a multiclass classifier, as STD-AA does.

However, the experimental results show STD-Bin to be inferior to STD-AA, as clear from both Figure 5.5 and Table 5.4. This is in line with the results of the preliminary experiments that lead us to not perform AV via LazyAV or StackedAV; see Section 5.3.3 for details.

In sum, given that STD-Bin is not a serious contender, the same considerations on the superiority of DV-based methods over standard methods made in Section 5.5.2.1 for AA also apply to AV.

<sup>14</sup>Note that, differently from our SAV experiments in Section 5.5.1, we here report experiments also for the combination <PAN11, 25, 50> since here we are only considering the closed-set setting, and since in PAN11 there are at least 25 authors with 50 documents; see also Footnote 9 in Section 5.5.1.1.

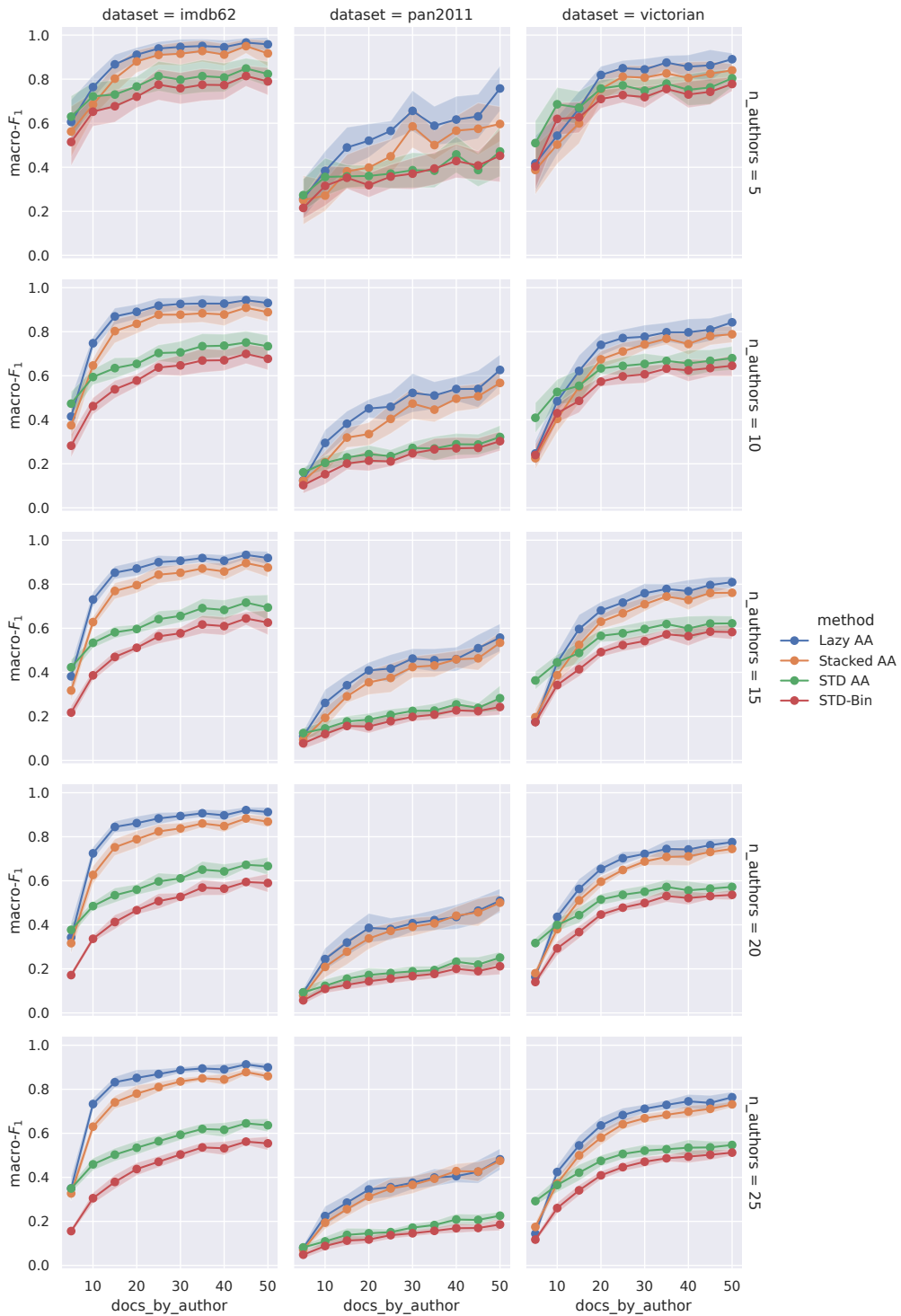


Figure 5.5: Extrinsic evaluation of DV: results on closed-set AA, with  $F_1$  as evaluation metric, for the IMDB62, PAN11, and VICTORIAN datasets.

### 5.5.3 Efficiency

As seen in the previous sections, the improvements in performance obtained by DV-based methods with respect to methods based on standard vectorial representation can be attributed to the increase in the number of training examples resulting from pairing documents. However, this can be expected to come at a computational cost. In this section, we compare the actual cost of DV-based methods with that of methods based on standard representation.

Specifically, we first analyze the computational cost of the different algorithms we employ in this project (Section 5.5.3.1) and show the actual timings clocked for our experiments (Section 5.5.3.2), proving that DVs do indeed occur in higher computational costs than methods employing the standard vector representation. Therefore, in Section 5.5.3.3 we argue that such higher costs should not be reduced at the expense of accuracy.

### 5.5.3.1 Efficiency analysis

Let  $n = |\mathcal{L}|$  be the number of training documents. We assume that the total cost of training a classifier is bounded by some function  $f$  on the number  $n$  of training documents, a cost which depends on the learning algorithm and its implementation; in other words, this total cost is  $O(f(n))$ , where we can safely assume  $f(n)$  to grow faster than, or equal to  $n$ . We take the number of features as constant, therefore it does not impact our analysis of efficiency. We also assume that the classification of a document requires constant time, thus being  $O(1)$ .

The cost of training the DV-Bin classifier of Section 5.3.1 comes down to the cost of generating the  $n(n-1)/2$  pairs, which is  $O(n^2)$ , plus the cost of training a classifier using  $n(n-1)/2$  DVs, which is  $O(f(n^2))$ . In practice, and in order to keep the computational burden under reasonable bounds, we only generate a fixed number of examples (i.e., we avoid generating all pairs first and discarding some of them later). Let  $n = mq$ , with  $m$  the number of authors and  $q$  the number of documents per author; we generate all  $mq(q-1)/2$  pairs of type Same and as many pairs of type Different, thus ending up with  $mq(q-1)$  documents, which has a cost  $O(mq^2) = O(nq)$ . In addition, again, the cost of training the classifier from the  $mq(q-1)$  documents is  $O(f(nq))$ . Since we assume  $f(n)$  to grow faster than or equal to  $n$ , the total cost of generating a DV-Bin classifier is  $O(f(nq))$ . At classification time, we only need to compute the absolute difference between two vectors and invoke the classifier; for most classifiers (and for LR in particular) this cost can be considered constant.

As a lazy algorithm, LazyAA does not involve any real training phase. However, it seeks for the optimal value of  $k$ , and this entails pre-computing a matrix of distances (which is done only once and is  $O(n^2)$ ), plus sorting, for each of the  $m$  authors and for each of the  $n$  training documents (see Equation 5.8), the  $q$  training documents by this author (which is  $O(q \log q)$ ). Altogether, this entails a total cost of  $O(n^2 + mnq \log q) = O(n^2 \log q)$ . At classification time, for both the AA and the AV settings), we only need to sort, for each of the  $m$  training authors, the  $q$  training documents by this author, which has a cost  $O(mq \log q) = O(n \log q)$ .

Concerning StackedAA, training the system entails training a DV-Bin classifier (with a cost of  $O(f(nq))$ , as already seen), creating the projections  $\phi(x)$  for each of the  $n$  training documents (with a cost  $O(n^2)$ , since creating one such projection has a cost  $O(n)$ ) and training the metaclassifier on the  $n$  vectors  $\phi(d)$  thus generated (which has a cost  $O(f(n))$ ); the total cost of training the system is thus the larger of  $O(n^2)$  and  $O(f(nq))$ . At classification time, we need to generate the representation  $\phi(d)$  of the test document, which has a cost  $O(n)$ , and to invoke the meta-classifier, which we can assume to require constant time.

The IM method does not properly carry out a training phase, but incurs the cost of optimising the  $\sigma^*$  parameter, which consists of carrying out  $t$  rounds of test, with  $t$  being a user-defined parameter. The computational cost of testing whether two documents have been written by the same author or not entails computing, for each of the  $n$  training instances, the similarity with each test document, which is  $O(n)$ , plus sorting by similarity in order to choose the impostors, which is  $O(n \log n)$ ; this means that the total cost is  $O(n \log n)$ . IM then performs  $k$  rounds of bagging trials with respect to each of the  $i$  impostors, which adds a cost  $O(ki)$  if we assume the similarity function to be computed in constant time.

Table 5.5 summarizes all the costs explained here.

Table 5.5: Computational cost of the algorithms discussed in this chapter. Table 5.6: Size of the datasets used for the efficiency test.

	Tasks	Training	Test
STD	AV, AA	$O(f(n))$	$O(1)$
DV	SAV	$O(f(nq))$	$O(1)$
LazyAA	AV, AA	$O(n^2 \log q)$	$O(n \log q)$
StackedAA	AV, AA	$\max\{O(n^2), O(f(nq))\}$	$O(n)$
IM	SAV	$O(n \log n)$	$O(n \log n)$

Dataset	$ \mathcal{L} $	$ \mathcal{L}_{\mathcal{P}} $	$ \mathcal{U} $
IMDB62	1,000	49,000	6,000
PAN11	1,000	49,000	463
VICTORIAN	1,000	49,000	7,937
ARXIV-SAV	518	5,784	255
ARXIV-AA	1,028	11,106	441

### 5.5.3.2 Timings

As for the experiments showed in Figure 5.2 and 5.4, we report actual timings clocked for  $m = 20$  and  $q = 50$  in the case of the IMDB62, PAN11, and VICTORIAN datasets, and for the entire dataset in the case of ARXIV. The variables that influence the analysis include the number of training documents ( $|\mathcal{L}|$ ), the number of pairs generated by DV ( $|\mathcal{L}_{\mathcal{P}}|$ ), and the number of test documents ( $|\mathcal{U}|$ ). Recall that  $|\mathcal{L}_{\mathcal{P}}|$  depends on the number of Same pairs that can be generated, which is fixed and amounts to  $20(50 \cdot 49)/2=24,500$  for IMDB62, PAN11, and VICTORIAN, and which is variable and depends on the random split for ARXIV (we report the value averaged across 10 runs). The values are summarised in Table 5.6 for convenience. Recall that the number of test pairs in SAV tasks is fixed for all datasets and is equal to 1,000. Note that the ARXIV dataset is split differently for SAV and AA since, although we used the entire dataset in both tasks, in the former we held half the authors out for composing the open set.

All times refer to computations carried out on the same machine, equipped with a 12-core processor Intel Core i7-4930K at 3.40GHz with 32 GB of RAM, under Ubuntu 18.04. All methods run on CPU and are implemented using `scikit-learn` and `SciPy` (see Appendix D). We have parallelised all parallelisable steps, both in training and test, for all algorithms.

The upper part of Table 5.7 reports the average time each method requires to complete the SAV task, both in terms of training time and testing time for each dataset. The STD-CosDist method, which uses standard vectors to compute the cosine distance, is much faster than any competing method, both in terms of training times and test times. This is due to the fact that cosine can be computed very quickly, and that the classifier operates on one single feature. At training time, both DV-Bin and IM are computationally much more expensive, with neither one being clearly better than the other.<sup>15</sup> However, DV-Bin is much faster than IM at classification time, and costs no more than a few seconds to accomplish the 1,000 SAV computations, comparably to STD-CosDist. IM, on the contrary, requires much more time, and its testing times are higher than its training times.

The lower part of Table 5.7 reports the average time each method requires to complete the AA Task. It is immediately evident that, at least on IMDB62, PAN11 and VICTORIAN, the two most expensive methods are the ones based on DVs, both at training time and at classification time (though neither one is systematically better or worse than the other); the STD method is almost always the fastest. The reason for this high computational cost of the DV-based methods is that, despite the fact that DV-Bin proved very fast at classification time in SAV, both LazyAA and StackedAA invoke DV-Bin multiple times: they require computing, for all training documents (in the training phase) and for all test documents (in the testing phase), the similarity (viewed as a posterior probability computed by DV-Bin) with each training document. This has an important impact both in the training phase and in the testing phase. Although the increase in training time with respect to the SAV task is not marked for the other datasets, for ARXIV the increase is substantial, because in this case the training set is twice as large as that for SAV (see

<sup>15</sup>Recall that, for IM, by “training” we mean the search for the optimal value of parameter  $\sigma^*$  by using the training set, since IM does not properly perform any training.

Table 5.7: Training and testing times (in seconds) clocked when solving the SAV and AA tasks. **Boldface** and underlining indicate the fastest and slowest methods for each task and each dataset, respectively.

		IMDB62		PAN11		VICTORIAN		ARXIV-SAV	
		Train	Test	Train	Test	Train	Test	Train	Test
SAV	DV-Bin	<u>438.3</u>	1.4	173.7	<b>0.7</b>	<u>870.2</u>	2.5	49.8	0.3
	STD-CosDist	<b>6.7</b>	<b>0.3</b>	<b>3.5</b>	0.8	<b>11.9</b>	<b>0.3</b>	<b>0.7</b>	<b>0.2</b>
	IM	271.9	<u>625.1</u>	<u>247.6</u>	<u>455.7</u>	283.2	<u>651.3</u>	<u>225.2</u>	<u>224.6</u>
AA	LazyAA	460.2	<u>955.5</u>	228.3	<u>61.8</u>	917.4	1232.6	<b>145.0</b>	66.1
	StackedAA	<u>480.3</u>	942.8	<u>267.3</u>	61.2	<u>955.4</u>	<u>1235.9</u>	250.4	<u>66.5</u>
	STD-AA	<b>156.0</b>	<b>0.2</b>	<b>157.6</b>	<b>0.1</b>	<b>202.9</b>	<b>0.5</b>	<u>483.8</u>	<b>0.1</b>

Table 5.6). The penalty paid during the testing phase is instead always evident: in some cases (IMDB62 and VICTORIAN), testing times even surpass training times, which can be explained by the fact that those datasets contain the largest test sets (6,000 and 7,937 instances, respectively). Somehow surprisingly, though, the variants based on DV-Bin are trained faster than STD in ARXIV; the reason for this lies in the number of authors involved, which in this dataset is the largest ( $m = 100$ ). STD thus needs to train 100 binary classifiers on a document-by-feature matrix of  $O(10^5)$  dimensions, while DV-based variants need to train only one binary classifier in order to discern between Same or Different; note also that in this case the number of pairs generated is comparatively smaller than for other datasets. The rest of the work that DV-based methods undertake is on a matrix of posterior probabilities that has just  $|\mathcal{L}|=1,028$  dimensions in the case of ARXIV; training 100 binary classifiers in StackedAA is thus much faster than with STD.

To conclude, DVs bring about substantially higher computational costs than the standard representation, both at training time and at testing time.

### 5.5.3.3 Are the costs incurred by Diff-Vectors tolerable?

The conclusions reached in Section 5.5.3.2 bring up the issue whether we should look for simplifications of our algorithms that cut down on training and/or test time, maybe at the cost of some reduction in accuracy.

One example is the use of StackedAA for performing AV. In the current setting, in order to perform AV (i.e., to choose, for a disputed document  $d$ , between a candidate author  $A_i$  and its complement  $\bar{A}_i$ ), we first invoke StackedAA in order to decide who among the authors of  $\mathcal{A} = \{A_1, \dots, A_i, \dots, A_m\}$  is the true author of  $d$ , and then we assign  $d$  to  $A_i$  if StackedAA predicts  $A_i$  to be the author of  $d$ , or to  $\bar{A}_i$  if StackedAA predicts an author in  $\{A_1, \dots, A_{i-1}, A_{i+1}, \dots, A_m\}$  to be the author of  $d$ . In other words, in order to solve the (binary) AV problem, we first invoke a multiclass algorithm (StackedAA) and then convert its multiclass decision into a binary decision. A much more efficient way would be to use StackedAV (see Section 5.3.3 for details), which directly solves a binary problem by means of a binary algorithm. However, as discussed in Section 5.3.3, preliminary experiments had shown this “direct” method to be less accurate than the “indirect” method we adopt. Other simplifications of LazyAA or StackedAA that we have come up with are, as in the case above, more efficient but less accurate.

Hence, there is a tradeoff between efficiency and accuracy, as it is often the case in classification. We have decided to stick to our methods, in the forms described in Sections 5.3.2.1 and 5.3.2.2, and to avoid the simplifications discussed above, because we think that accuracy should not be compromised in favor of efficiency in AId settings, for three reasons. The first reason is that, in real AId cases, especially within the cultural heritage domain, increased classification times are usually tolerable, because such cases typically do not involve many unlabelled documents. Indeed, there is often a single unlabelled document, of extremely high value, that we need to make a prediction for (e.g., a text of literary value,

Table 5.8: Results, in terms of  $F_1^M$ , obtained by applying AA methods to “native” AV problems. The notational conventions are the same as in Table 5.2.

	IMDB62			PAN11			VICTORIAN			ARXIV		
	mean	std	ttest	mean	std	ttest	mean	std	ttest	mean	std	ttest
LazyAA	0.287	0.044		0.166	0.019		0.205	0.010		0.234	0.037	
StackedAA	0.664	0.062	**	0.278	0.056	**	0.619	0.044	**	0.365	0.075	**
STD-Bin	<b>0.683</b>	0.045		<b>0.285</b>	0.028		<b>0.639</b>	0.042		<b>0.432</b>	0.097	

or an anonymous letter), and in this case issuing a prediction in milliseconds or in minutes does not make a big difference. The second reason is that the decisions of an AId system are often of paramount importance (typical examples are again its applications in cultural heritage, but also in cybersecurity and forensics), so no user would prefer to increase the risk of a misidentification for the sake of efficiency. The third reason is that classifier training is performed only once, and the times reported in Tables 5.7 are plausible for most application contexts. Note also that training documents are scarce in most AId applications, which means that scenarios in which the training documents are many more than in our datasets are unfortunately infrequent.

All this indicates that reduced (training and/or testing) efficiency is not a critical issue in AId, and that this reduced efficiency is tolerable if it leads to higher accuracy.

#### 5.5.4 Can we use Diff-Vectors for “natively binary” AV problems?

In the previous sections, we always test DVs in situations in which, at training time, we assume to know who among the  $n$  authors in  $\mathcal{A}$  has written the training documents; that is, we have recast SAV, AA and AV in terms of a multiclass task. In this section, we experimentally analyse the suitability of DVs for AV in a different setting: we assume that all we know about a certain training document is whether it has been written by the author of interest or not, that is, whether this document is a positive example or a negative example with respect to a binary classification scheme.

To this aim, we randomly draw  $m = 10$  authors for each dataset, and perform an AV experiment for each author  $A_i$ , in which we take  $A_i$  as the positive class and the rest of the authors (grouped together) as the negative class, and where we employ an AA method (i.e., one that was originally devised for tackling arbitrary values of  $n$ ) for the particular case of  $n = 2$ . Formally, given  $\mathcal{A} = \{A_1, A_2, \dots, A_{10}\}$ , we generate a binary setting  $\mathcal{A}'_i = \{A_i, \bar{A}_i\}$ , and we do this for all authors by letting  $i$  vary in the range  $\{1, \dots, 10\}$ . In each of these experiments, we take  $q = 50$  documents for each author in  $\mathcal{A} = \{A_1, A_2, \dots, A_{10}\}$  in all datasets, except for ARXIV, for which we take all the documents available for the author. We repeat the entire process 10 times with different random seeds and report results averaged across all experiments. The results reported in Table 5.8 show that, in such a setting, DVs do not bring about any benefit.

This was to be expected, since DVs bring useful additional evidence to the learning process for AV only when we have access to the entire labelling information, i.e., when author  $A_j$  can help improve classification for author  $A_i$  indirectly, by strengthening the internal SAV function with additional instances of the class **Same** that come from documents written by  $A_j$  (see Section 5.2.1). This is not possible in a pure binary setting, since the negative class is not homogeneous (i.e., it does not represent the production of one single author, but the production of many authors that have been mixed together), and thus cannot be leveraged to generate positive instances for the class **Same**. For similar reasons, we cannot generate instances of **Different** by simply picking two documents from the negative class, since those could have been written by the same (unknown) author. Given this, we are left with the possibility to generate  $q(q - 1)/2$  instances of **Same** only from pairs of instances from the positive class  $A_i$ , and  $q^2(m - 1)$  instances of **Different** by generating pairs in which one document has been written by  $A_i$  and the other by  $\bar{A}_i$ . Since the positive evidence for the surrogate SAV problem (class **Same**) comes exclusively from

Table 5.9:  $F_1^M$  and  $F_1^\mu$  results obtained on the AA task with a different supervised learning method (SVM). The LR results are included for comparison purposes, and are repeated from Section 5.5.2.1.

	Learner	Method	IMDB62	PAN11	VICTORIAN	ARXIV-AA
$F_1^M$	LR	StackedAA	<b>0.8596</b> $\pm$ 0.0163	<b>0.4754</b> $\pm$ 0.0256	<b>0.7315</b> $\pm$ 0.0131	<b>0.6386</b> $\pm$ 0.0277
		STD-AA	0.6365 $\pm$ 0.0295	0.2259 $\pm$ 0.0161	0.5471 $\pm$ 0.0172	0.3741 $\pm$ 0.0178
	SVM+LRI	StackedAA	<b>0.8270</b> $\pm$ 0.0226	<b>0.4313</b> $\pm$ 0.0323	<b>0.6549</b> $\pm$ 0.0276	<b>0.5655</b> $\pm$ 0.0227
		STD-AA	0.6213 $\pm$ 0.0335	0.2095 $\pm$ 0.0263	0.4722 $\pm$ 0.1236	0.3177 $\pm$ 0.0220
$F_1^\mu$	LR	StackedAA	<b>0.8599</b> $\pm$ 0.0158	<b>0.5546</b> $\pm$ 0.0331	<b>0.7509</b> $\pm$ 0.0163	<b>0.6395</b> $\pm$ 0.0236
		STD-AA	0.6406 $\pm$ 0.0286	0.2682 $\pm$ 0.0231	0.5795 $\pm$ 0.0276	0.4045 $\pm$ 0.0213
	SVM+LRI	StackedAA	<b>0.8233</b> $\pm$ 0.0217	<b>0.5066</b> $\pm$ 0.0376	<b>0.6928</b> $\pm$ 0.0305	<b>0.5889</b> $\pm$ 0.0184
		STD-AA	0.6381 $\pm$ 0.0307	0.2660 $\pm$ 0.0313	0.5183 $\pm$ 0.1230	0.3857 $\pm$ 0.0198

the positive class of the AV problem (author  $A_i$ ), there is no real information gain with respect to using standard representation. Indeed, we observe a degradation in performance of both variants with respect to the adoption of standard vector representation (even though it is not statistically significant).

For this reason, we conclude that, in AV settings, DV-based methods should be used only in situations in which we have access to the entire class label information. Luckily enough, access to the entire class label information is something that characterises most scenarios in which AV is to be applied, since, when investigating whether document  $d$  is indeed by author  $A^*$  or not, it makes sense to generate a training dataset in which negative instances are known to be by authors close (in a stylistic sense) to  $A^*$ , and we can do this only by knowing who the author of each document is.

### 5.5.5 Results with a different learner

In this section, we discuss some additional experiments, which we run in order to check whether the DV-based methods are superior to methods based on standard vectors also when using a learning algorithm different from LR. In fact, like the standard representation, the DV-based representation is learner-independent, i.e., it can be used in connection with any (supervised or unsupervised) learning method.

For these experiments, in which we compare the StackedAA DV-based method with the “standard” STD-AA method (see Section 5.5.2.1), indeed we switch to SVMs, since they are frequently used in the AAn field (see Section 2.3), where they have usually shown very good performance. Similarly to LR, we use the implementation of `LinearSVC` available from the `scikit-learn` library (see Appendix D). In order to speed up the computation, we simply rely on default hyper-parameters and perform dimensionality reduction using “lightweight random indexing” [205], projecting the original space onto a 1,000-dimensional space. This attempt at reducing the computational cost of the process is justified by the fact that SVMs are not probabilistic classifiers, and thus their outputs need to be converted into posterior probabilities; this conversion is done via calibration, a process that incurs the additional cost of performing model selection via cross-validation. For this, we set the number of folds to 3 and adopt Platt scaling as our calibration method.

The  $F_1^M$  and  $F_1^\mu$  results reported in Table 5.9 are obtained by running the corresponding setup for  $m = 25$  randomly chosen authors, for which we randomly choose  $q = 50$  training documents per author (except for the ARXIV dataset, for which we use all data available). We report the mean results, along with their standard deviation, across 10 runs performed with different random seeds. As evident from these results, the conclusions drawn from the LR experiments (see Section 5.5.2.1) are confirmed: the StackedAA DV-based method is largely superior to the “standard” STD-AA method. Indeed, the former outperforms the latter on all four datasets and for both evaluation measures, always by a large margin.

As a final note, we recall from Section 5.5.2.2 that the  $F_1^M$  results of an AA experiment also count as an evaluation on the AV task, which indicates that DV-based methods are superior not only in terms of AA, but also in terms of AV, also when using SVM as the learning algorithm.

## 5.6 Discussion

In this chapter, we have discussed and analyzed the implications of the use of Diff-Vectors (DVs) in AId tasks. A DV is a vector that represents a pair of documents in such a way that the value of a feature in the DV is the absolute difference between the relative values of the feature in the original two documents. DVs were originally introduced by Koppel and Winter [176], but neither them nor other authors systematically studied the implications of this methodology; we carried out this investigation here. We pursued this evaluation in order to tackle what in this Thesis we define as **Problem 2**, that is, the problem of having to deal with datasets of extremely limited size, which is rather common in AAn settings in general, and within the cultural heritage domain in particular.

In order to compare DV-based methods with their counterparts based on “standard” vectors, we have carried out experiments on three AId tasks (SAV, AA, AV). In particular, DVs are naturally geared towards solving the SAV task; however, we have shown that both (closed-set) AA and AV can be recast in terms of SAV, for which we have presented two original algorithms (LazyAA and StackedAA).

Our experiments show that the DV-based representation is advantageous in most cases, since it brings about substantially increased effectiveness. As we have argued, these benefits derive from the fact that, in many cases, DV-based methods may exploit more training data than methods based on standard vectors (see Section 5.2.1), and that they may make training more robust also when the above is not the case (see Section 5.2.2). Indeed, the experiments also show that DVs bring about substantial improvements especially in low-resource AId tasks, i.e., in tasks characterised by small quantities of training data.

We have also presented an extensive comparative analysis of the efficiency of these methodologies, both by studying their computational complexity and by clocking the actual experiments. This study confirms that, as expected, the DV-based methodology is computationally more expensive; however, as we argue in detail, the additional computational cost is tolerable, especially since classification efficiency is rarely a primary concern in practical application scenarios of AId.

In future work, it would be interesting to study “diff-functions” other than the absolute difference of the feature values of the two documents, by testing the possibility of dynamically learning such functions from data, in the style of Moreo et al. [207]. Other aspects worth exploring include testing DVs in a broader range of AAn tasks, such as AP tasks.



## Chapter 6

# An attempt to unveil authorship forgery via data augmentation

Training and employing an effective AAn classifier can be very taxing, or even impossible, if an “adversary” is at play, i.e., when a human or an automatic process actively tries to mislead the classification. We can distinguish between two main variants of this adversarial setting [54]. In the first case, called “obfuscation”, the authors themselves try to conceal their writing style, in order not to be recognized. This can be done manually, but nowadays there are specific automatic tools for achieving it; for example, by applying sequential steps of machine translation [96]. In the second case, called “imitation”, a forger (the “imitator”) tries to replicate the writing style of another specific author. While the former case can be considered as possibly less harmful (a writer might simply be interested in preserving their privacy, without necessarily harbouring any malevolent intent), the latter case is inherently illicit (unless the imitated author gives explicit consent, or in the case of artistic tributes, provided they are accompanied by a statement openly acknowledging the intent).

Our cultural heritage is indeed filled with countless examples of historical documents of questioned authorship, often caused by supposed forgeries or false appropriations [76, 199, 216, 256, 284, 288]. Moreover, due to the recent significant advances in language modelling, powerful NNs are now able to autonomously generate “coherent, non-trivial and human-like” text samples [95, p. 1], that can be exploited as fake news or propaganda [95, 248]. Indeed, it has been shown that many AId systems can be easily deceived in adversarial contexts [54, 236]. This has given rise to a discipline known as “adversarial authorship” (or “adversarial stylometry”, or “authorship obfuscation”) devoted to study specific techniques able to fool AId systems [16, 41, 96]. Similarly, research such as the work by Afroz et al. [12] investigate the possibility to detect stylistic deception in written documents, although without framing it as an authorship task.

A seemingly straightforward solution to this problem involves enriching the training set of the classifier with representative texts from the forger, enabling the training algorithm to identify descriptive patterns that effectively detect adversarial examples. Unfortunately, this strategy is generally unfeasible, as in most cases we could not expect to have any such examples.

In this chapter, we investigate ways for improving the performance of an AV classifier by augmenting its training set with *synthetically* generated examples that mimic the style of the author that the classifier tries to identify, tackling the **Problem 3** of this Thesis. After surveying the related literature in Section 6.1, we present our method in Section 6.2, describing the generator architectures we employ and their training strategies, and the learning algorithms that we employ for the AV classifier. In particular, we explore various generator architectures, including a GRU model [72], a standard transformer [291], and

the popular GPT system (in its shallower variant, DistilGPT2 [238]). We also adopt two distinct training strategies for the generators: one draws inspiration from standard Language Models (LMs) where the generator learns to emulate the writing style of a specific author, while the other takes cues from Wasserstein Generative Adversarial Networks (WGANs) by training a generator to exploit the weaknesses of the classifier. Moreover, we use two learning algorithms for the AV classifier: SVM and CNN (see Section 2.3 and Section 2.4). In Section 6.3, we detail the experimental setting, including the five datasets we use (three of them collected specially to simulate an adversarial setting), and our experimental protocol. In Section 6.4 we present and comment on the results of our experiments; despite our efforts, the results we have obtained seem to indicate that data augmentation (at least with the techniques we study) is not always beneficial for the task of AV. Finally, Section 6.5 wraps up, offering some final remarks and pointing to some possible avenues for future research.

The research presented in this chapter is detailed in an extended abstract and an article by Corbara and Moreo [3, 4].<sup>1</sup>

## 6.1 Related work

The use of data augmentation for improving classification performance is not a new idea, neither in the text classification field nor in the AAn field. Researchers have explored various techniques for generating synthetic samples, such as the random combination of real texts [49, 280], the random substitutions of words with synonyms [305], and using the interrogation of LMs [173]. Similarly, adversarial examples are generated by a process that actively tries to fool the classification [109], and have been extensively used to improve the training of a classifier for many text classification tasks in general, and for counteracting adversarial attacks in particular. For example, Zhai et al. [304] feed the learning algorithm with (real) texts that have been purposefully obfuscated, in order to make the classifier robust to obfuscation.

Unlike these works, we do not automatically modify pre-existing texts; instead, we employ various generation algorithms to create new samples, simulating the deceitful actions of an adversary. In particular, one generative technique we explore is the so-called Generative Adversarial Network (GAN), which was first introduced by Goodfellow et al. [108]. The GAN architecture is made of two components: a Generator ( $G$ ) that produces synthetic (hence fake) examples, and a Discriminator ( $D$ ) that classifies examples as “real” (i.e., coming from a real-world distribution) or “fake” (i.e., generated by  $G$ ). Both components play a min-max game, where  $D$  tries to correctly spot the examples created by  $G$ , while  $G$  tries to produce more plausible examples in order to fool  $D$ .

Although the GAN strategy has excelled in the generation of images [157], its application to text generation has proven rather cumbersome. One of the main reasons behind this is the discrete nature of language: choosing the next word in a sequence implies picking one symbol from a vocabulary, an operation that is typically carried out through an argmax operation, which blocks the gradient flow during backpropagation. Some strategies have been proposed to overcome this limitation. One idea is to employ a reinforcement learning approach, where the feedback from  $D$  acts as reward [301], although this strategy has been found to be inefficient to train [86]. An alternative idea is to use the Gumbel-Softmax operation to obtain a differentiable approximation of one-hot vectors [177], or to directly operate with the continuous output of the generator (i.e., refraining from choosing specific tokens); some examples include the encoding of real sentences via an autoencoder [86], or the creation of a word-embedding matrix for real sentences [306].

In this prospective, the work by Hatua et al. [126] bears some similarities with our methodology:

---

<sup>1</sup>The code to replicate these experiments is available at: [https://github.com/silvia-cor/Authorship\\_DataAugmentation](https://github.com/silvia-cor/Authorship_DataAugmentation).

they employ a GAN-trained generator to produce new examples, which are labelled as negatives and then added to the classifier training dataset. Their experiments indeed demonstrated the benefits of data augmentation for the fact-checking task.

Within the AId field, the work by Manjavacas et al. [197] explores the idea to use data augmentation to boost the performance of an AA classifier, employing a RNN as language model. They indeed report a slight increase in the attribution performance of the system; however, they perform a very narrow experimentation, without significance testing, and do not tackle the problem of forgery, limiting their investigation to a single non-adversarial setting.

## 6.2 Methodology for synthesizing forged documents

In this Thesis, we frame the AV task as a binary text classification problem in which the goal is to determine whether a given document  $d$  is authored by a specific candidate author (positive class  $A^*$ ), or if it is instead the work of any other author (collective negative class  $\bar{A}^*$ ); see also Section 1.1. The union of all the available texts with corresponding labels define the training set  $L$  that we use for generating a classifier  $h$  via an inductive algorithm.

We propose to enhance the classifier performance with the addition of adversarial training examples, i.e., textual examples specially generated to imitate the author  $A^*$ . The generated examples are labelled as negative examples ( $\bar{A}^*$ ) and added to  $L$  with the scope of generating an improved classifier  $h^*$ . Note that, unlike works such as the one by Jones et al. [149], we do not employ the newly generated examples as synthetic positive instances (in our case: for the class  $A^*$ ); otherwise, the classifier would learn to label fraudulent instances as  $A^*$ , which is not our aim.

This process is sketched in Figure 6.1.<sup>2</sup> As shown in the diagram, we explore three different generator architectures (GRU, TRA, and GPT) that we discuss in Section 6.2.1, and that we train using two different learning algorithms (LMtr and GANtr) that we discuss in Section 6.2.2. Concerning the underlying classifier of our AV system, we experiment with two learning algorithms (SVM and CNN) that we discuss in Section 6.3.2.

### 6.2.1 Generator architectures

In order to generate the adversarial examples, we experiment with three alternative generator architectures of increasing level of complexity:

- **GRU.** Ezen-Can [93] has shown that simpler recurrent models (specifically: LSTM) tend to outperform more sophisticated models (specifically: BERT) when the training data is small, since simpler models have fewer parameters and are thus less prone to overfitting. Given that data scarcity is characteristic of many AV settings, we consider the Gated Recurrent Unit (GRU) [72] model, a simplified variant of LSTM. We set our model with 2 unidirectional GRU-layers of 512 hidden units each, followed by a linear layer with a ReLU activation function.<sup>3</sup>
- **TRA.** We consider the original transformer introduced by Vaswani et al. [291], that replaced a recurrently-handled memory in favor of fully-attention layers. We set our model with 2 encoder layers of 512 hidden dimensions each and 4 attention heads, followed by a linear layer with a ReLU activation.<sup>4</sup>

---

<sup>2</sup>Icons made by Vitaly Gorbachev on Flaticon: <https://www.flaticon.com/>.

<sup>3</sup>We use the PyTorch implementation of GRU: <https://pytorch.org/docs/stable/generated/torch.nn.GRU.html>.

<sup>4</sup>Our implementation relies on Pytorch's TransformerEncoder: <https://pytorch.org/docs/stable/generated/torch.nn.TransformerEncoder.html>.

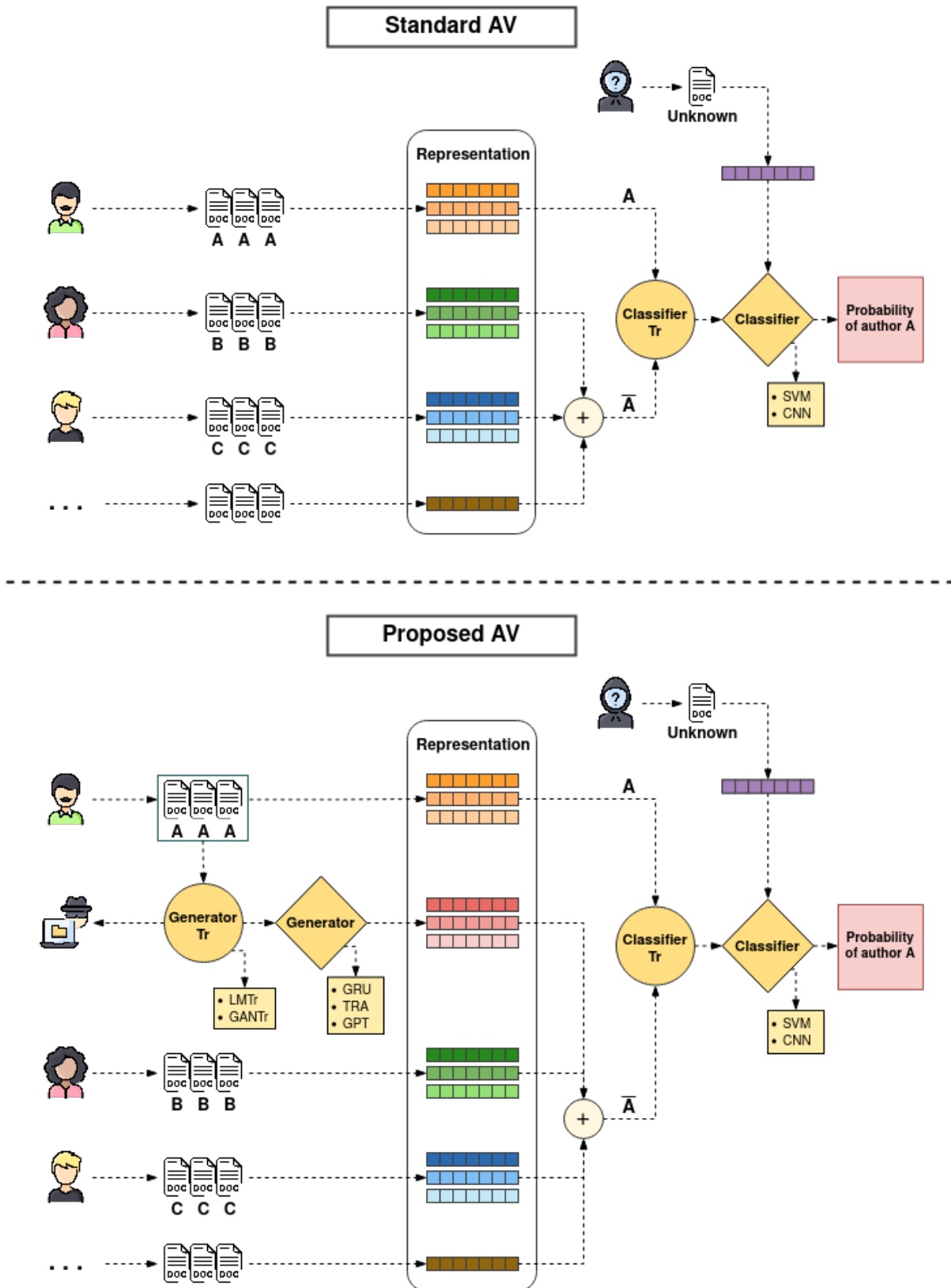


Figure 6.1: Upper: Flowchart of a standard AV method. Bottom: Flowchart of our proposed AV method, where representative examples of forgery are added to  $\bar{A}^*$ .

- **GPT.** The forefront in current generation models is held by very large LMs, with Chat-GPT 4 occupying the top position. Unfortunately, these models are huge in terms of number of parameters, and are typically not released to the scientific community, unless behind a pay-wall and an API. As a representative (though much smaller) model, we focus on GPT-2, a pre-trained unidirectional transformer [238] released by OpenAI, that became increasingly popular for its outstanding generation performance and general-purpose nature. Uchendu et al. [287] and Fagni et al. [95] assessed the quality of its generated texts, concluding they are often more human-like than those from other text generators. To limit the number of parameters and speed up the computation, while retaining the good quality of the original model, we employ DistilGPT2 [250], a smaller model (82M parameters versus the 124 M parameters of the standard version of GPT-2) that is trained via knowledge distillation on GPT-2. Note that this model outputs a multinomial distribution over the entire vocabulary at each generation step, and then selects the next token by using a sampling technique; we use the top- $k$  sampling, where the  $k$  most-likely next words are filtered, and the probability mass is redistributed among those  $k$  words (we set  $k = 50$ ). We enforce no repetitions within 5-grams.<sup>5</sup>

We explore different vector representations for the input and output of the generators:

- **One-hot encoding (1h):** a typical representation in which a vector of length  $|V|$ , where  $V$  is the vocabulary, contains exactly one “1”, whose index identifies one specific word in  $V$  (all other values are set to “0”). We use the DistilGPT2 word tokenizer in all our experiments, which gives rise to a vocabulary size of  $|V| = 50,257$  tokens. By GRU<sub>1h</sub> and TRA<sub>1h</sub> we denote the variants equipped with an input linear layer (without bias) that projects the one-hot representation onto 128 dimensions, hence densifying the representation (GPT is a more complex model and we do not explore the 1h variant). The output of these models is generated by a last linear layer of  $|V|$  dimensions. During the GANtr, we compute the Gumbel-Softmax distribution over the output in order to discretize the output (that determines the next word in the sequence) without interrupting the gradient.<sup>6</sup>
- **Dense encoding (emb):** we also experiment with a variant in which the generator produces embeddings of the same dimension of the ones used by the CNN method of Section 6.3.2; we call the three models GRU<sub>emb</sub>, TRA<sub>emb</sub>, and GPT<sub>emb</sub>, respectively.

### 6.2.2 Generator training

We explore two different strategies for training the generator architectures described above: one based on standard Language-Model training (hereafter: LMtr), in which the generator is trained to replicate the style of the author of interest, and another based on GAN training (hereafter: GANtr), where the generator plays the role of the forger that tries to fool the discriminator.

- **Language Model Training (LMtr).** A standard way in which LMs are trained comes down to optimizing the model to predict the next word in a sequence, for a typically large number of sequences. More formally, given a sentence  $w_1, \dots, w_t$  of  $t$  tokens, the model is trained to maximize the conditional probability  $\Pr(w_t|w_1, w_2, \dots, w_{t-1}; \Theta)$ , where  $\Theta$  are the model parameters. Such sequences are drawn from real textual examples, and could either come from a generic domain (thus optimizing a LM as representative of a language in general) or from a specific domain (thus optimizing the LM for a particular area of knowledge or task). We are interested in the latter case. Specifically, we draw sequences from texts that we know have been written by  $A^*$ , thus attaining

<sup>5</sup>We relied on the Huggingface’s `transformers` implementation: <https://huggingface.co/distilgpt2>.

<sup>6</sup>We rely on the PyTorch implementation: [https://pytorch.org/docs/stable/generated/torch.nn.functional.gumbel\\_softmax.html](https://pytorch.org/docs/stable/generated/torch.nn.functional.gumbel_softmax.html)

a LM that tries to imitate the author (i.e., that tries to choose the next word as  $A^*$  would have chosen). This idea has been shown to retain the stylistic patterns typical of the target author to a certain extent [149]. Of course, the main limitation of this strategy is the limited amount of sequences we might expect to have access to, since these are bounded by the production of one single author; such sequences might be very few when compared with the typical amount of information used to train LMs.

- **Generative Adversarial Network Training (GANtr).** The Wasserstein GAN (WGAN) approach [22] is based on a GAN architecture (see Section 6.1) that relies on the Earth-Mover (also called Wasserstein) distance as the loss function.<sup>7</sup> This loss is continuous everywhere and produces smoother gradients, something that has been shown to prevent the gradient-vanishing problem and the mode-collapse problem that typically affect the early stages of the GAN training, when the generator  $G$  still performs poorly. Gulrajani et al. [113] later developed WGANGP, that improves the stability of the training by penalizing the gradient of the discriminator  $D$  instead of clipping the weights (as proposed in the original formulation), which is the approach that we adopt here.<sup>8</sup> As the generator  $G$ , we explore the architectures (GRU, TRA, GPT) described in Section 6.2, while for the discriminator  $D$ , we employ a CNN-based classifier (the same CNN classifier that we describe in detail in Section 6.3.2).

## 6.3 Experimental setting

In this section, we present the setting of the experiments we have carried out: in Section 6.3.1 we describe the datasets we employ, while in Section 6.3.3 we describe the experimental protocol we follow.

### 6.3.1 Datasets

We experiment with five publicly available datasets, some of which are examples of a close-set setting (where the authors comprising the test set are also present in the training set), while others are instead open-set (where the authors comprising the test set are not necessarily present in the training set).

- **TWEEPFake.** See Appendix B.9 for the full description of this dataset. Since it would realistically be rather problematic to obtain a corpus containing human forgeries, we use this dataset as a reasonable proxy, where the forgery is made by a machine trying to emulate a human writer. In order to reproduce a more realistic setting, we only consider the documents produced by human users for our training and validation sets, but we keep all the documents in the test set so as to emulate an open-set problem.
- **EBG.** See Appendix B.10 for the full description of this dataset. We randomly select 10 authors and use all the examples of their writing to compose the train-and-validation set; we split it with a 90/10 ratio into training and validation sets, in a stratified fashion. All the obscured short essays form the test set instead, making it an open-set scenario. We use these documents in order to check the ability of the model to recognize the author of interest even in cases where they actively try to mask their writing.

---

<sup>7</sup>In the related literature, the discriminator underlying a WGAN is sometimes called “the critic” instead of “the discriminator”, since it outputs a confidence score in place of a posterior probability. This distinction is rather unimportant for the scope of our work, so we keep the term “discriminator” for the sake of simplicity.

<sup>8</sup>We use the PyTorch-based implementation available at: [https://github.com/eriklindernoren/PyTorch-GAN/tree/master/implementations/wgan\\_gp](https://github.com/eriklindernoren/PyTorch-GAN/tree/master/implementations/wgan_gp).

Table 6.1: Number of authors in training, and number of training, validation, and test examples in each dataset.

	#authors	#training	#validation	#test
TWEEPFake	15	3,099	331	761
EBG	10	800	89	270
RJ	10	598	67	161
PAN11	3	90	47	348
VICTORIAN	5	4,050	450	500

- **RJ.** See Appendix B.11 for the full description of this dataset. These texts are similar in nature to the ones in the EBG corpus, and we use them for analogous reasons. We randomly select 10 authors and split the train-and-validation set as in the EBG corpus, while keeping the whole test set, thus making it an example of an open-set scenario.
- **PAN11.** See Appendix B.6 for the full description of this dataset. We merge the three training sets into a single one (resulting in a training set with three authors), and we do the same for the validation and test sets. We use this dataset as an example of contemporary production “in the wild”, where the authors are realistically not trying to conceal their style, and are prone to all the mistakes and noise of digital communication. Moreover, the training set is rather limited (there are only two authors representing the  $\bar{A}^*$  class), and the resulting AV problems are open-set.
- **VICTORIAN.** See Appendix B.7 for the full description of this dataset. We use these documents as examples of literary production, where no author is presumably trying to imitate someone else’s style, nor conceal their own. We limit the dataset to 5 authors selected randomly with 1,000 chunks each (see below), in order to tackle the experimental setting where, unlike in the other datasets, there are few authors but a fair abundance of data. We divide the data into a training-and-validation set and a test set with a 90/10 ratio, and further divide the former into training and validation set with another 90/10 ratio, in a stratified fashion. This dataset is representative of a closed-set AV problem.

In each dataset, we split each document into non-overlapping chunks of 100 tokens (words and punctuation), and we discard final chunks of less than 25 words (not considering punctuation); we carry out this segmentation before the partitioning the dataset into a training, validation, and test set. We also exclude authors with less than 10 chunks in the training set. Table 6.1 shows the final number of training, validation, and test examples for each dataset.

### 6.3.2 Learners and features

For our AV system, we experiment with two classifiers, that we describe here: one is based on SVM and the other is based on CNN. Before describing the learning algorithms we employ, we present a set of features we extract that are commonly employed in the AAn literature.

- **Base features.** Similarly to what we do in Section 4.1.3.2 and Section 4.2.3.2, we define a set of features that we call BaseFeatures (from now on: BFs), comprised of features that have been proven useful in the related literature and are now considered standard in many AAn studies (for a discussion regarding these features, and more generally regarding feature design, see Section 2.1). In particular, we employ the following features:
  - **Function words:** the normalized relative frequency of each function word; we use the list provided for the English stopwords by the NLTK library (see Appendix D);
  - **Word lengths:** the relative frequency of words up to a certain length in the range  $[1, n]$ , where  $n$  is the longest word appearing at least 5 times in the training set;

- **POS-tags:** the normalized relative frequency of each Part-Of-Speech (POS) tag; we extract the POS-tags using the `spaCy` library (see Appendix D).
- **SVM classifier.** We consider a SVM-based classifier as our AV system, due to the good performance SVM have demonstrated in text classification tasks in general over the years [145], and in AAn tasks in particular [166]. We employ the `SVC` implementation from the `scikit-learn` package (see Appendix D). We optimize the hyper-parameters of the classifier via grid-search. In particular, we explore the parameter  $C$  in the log-space  $\{10^i\}_{i=0}^3$ , the kernel in the range  $\{linear, poly, rbf, sigmoid\}$ , and we explore whether to rebalance the class weights or not. After model selection, the SVM is then re-trained on the union of training and validation sets with the optimized hyper-parameters, and then evaluated on the test set.

As features, in addition to the BFs set, we also use the character  $n$ -grams, where  $n$  is in the range [1, 3]. Since these features result in high dimensionality, we employ the 10% most discriminating char  $n$ -grams in terms of Chi-square (see Appendix A.2).

- **CNN classifier.** We also experiment with an AV classifier based on the following CNN architecture. The input layer of the CNN has variable dimensions depending on the vector modality: the dense representations (`emb`) depend on the generator architecture of choice (GPT produces 768-dimensional vectors, while we set the output dimensions of GRU and TRA to 128), while the one-hot representations (`1h`) consist of  $|V| = 50,257$  dimensions that are projected into a 128-dimensional space by means of a linear layer (without bias). This is followed by two parallel convolutional blocks with kernel sizes of 3 and 5, respectively. Each convolutional block consists of two layers of 512 and 256 dimensions and ReLU activation. We then apply max-pooling to the resulting tensors and concatenate the outputs of both blocks. We also apply dropout with 0.3 probability, and the resulting tensor is then processed by a linear transformation of 64 dimensions with ReLU activation. For generator architectures for which we can derive “plain texts” (i.e., for all the `1h`-variants and GPT), we add a parallel branch that receives the relative BFs as inputs; these features are processed by two linear layers of 128 and 64 dimensions and ReLU activation. The resulting tensor is then concatenated with the output of the other branch, and the result is passed through a final linear layer with ReLU activation which produces a single value representing the confidence score of the classifier. The complete CNN model is depicted in Figure 6.2.

We train the network using the AdamW optimizer [191] with a learning rate of 0.001, a batch size of 32, and binary cross entropy as the loss function. In order to counter the effect of class imbalance (there are many more negatives than positives), we set the class weights to the ratio between negative and positive examples. We train the model for a minimum of 50 epochs and a maximum of 500 epochs, and apply early stopping when the performance on the validation set (as measured in terms of  $F_1$ ) does not improve for 25 consecutive epochs. Finally, we further train the model on the combination of training and validation sets for 5 epochs before the model evaluation.

### 6.3.3 Experimental protocol

For each dataset, we perform the experiments by iterating through rounds where each author is treated as the positive class  $A^*$ , while the remaining authors are considered as the negative class  $\bar{A}^*$ .

In each generation step, we generate  $n$  new examples, where  $n$  is set to 10 times the number of training chunks for  $A^*$ , up to a maximum of 1,000 new generated examples. As prompt for each new generation, we use the first 5 tokens from a randomly selected training chunk by  $A^*$ ; the generated text has the same

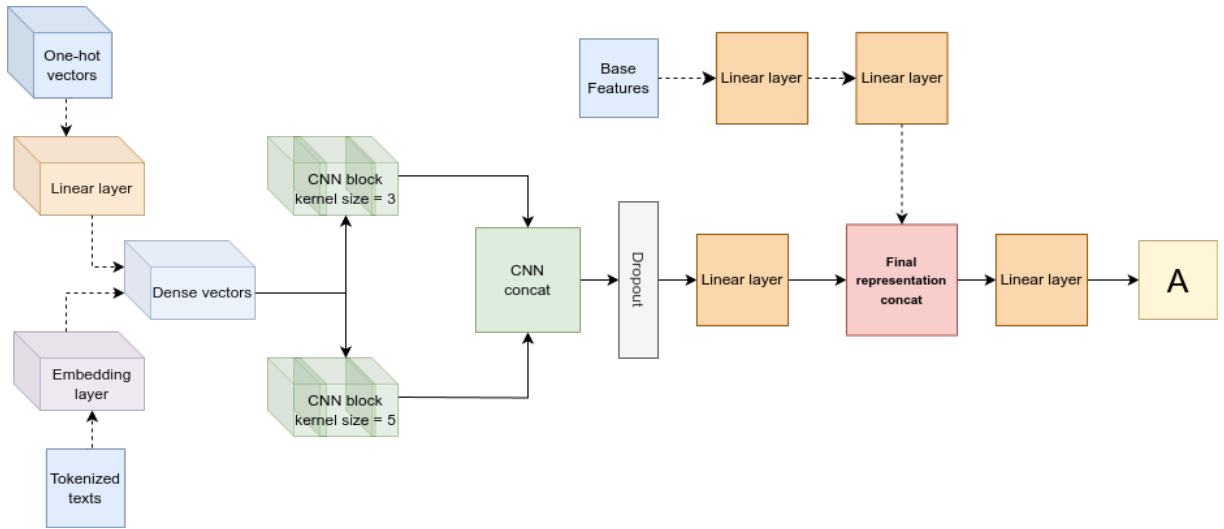


Figure 6.2: The CNN classifier architecture. The dotted lines represent alternative branches.

length as the original chunk by  $A^*$ . Once the  $n$  examples have been generated, they are labelled as  $\bar{A}^*$  and added to the training set, which is now used to train the classifiers discussed in Section 6.3.2.

We denote the classifiers trained with data augmentation with the following nomenclature:  $C + G_E^T$ , where  $C$  is a classifier from Section 6.3.2,  $G$  is a generator architecture from Section 6.2.1,  $T$  is a training strategy for the generator from Section 6.2.2, and  $E$  is an encoding type (**1h** or **emb**, see Section 6.2). For additional details on each combination, see Table 6.2. Note that the natural baseline for any augmentation setup  $C + G_E^T$  is  $C$ , i.e., the same classifier trained without the newly generated examples.

We measure the performance of the classifier in terms of the well-known  $F_1$ <sup>9</sup> and  $K$  metric (see Appendix A.1). We report the average in performance for both metrics across all experiments per dataset.

Since our goal is to improve the classifier performance with the addition of synthetically generated examples, we compute the relative improvement of the classifier trained on the augmented dataset compared to an instance of the same classifier trained solely on the original training set (thus excluding adversarial examples). We also compute the statistical significance of the differences in performance via the McNemar’s paired non-parametric statistical hypothesis test [200]. To this aim, we convert the outputs of the two methods (with and without augmentation) into 1 (correct prediction) or 0 (wrong prediction). We take 0.05 as the confidence value for statistical significance.

## 6.4 Results

Table 6.3 reports the results we have obtained for the different combinations of classifiers and generation procedures. Note that some combinations are missing; in particular, SVM are not combined with the **emb**-variants of GRU or TRA since, in those cases, and in contrast to GPT, we are not able to generate plain texts from which BFs can be extracted.

Statistical tests of significance reveal that data augmentation yields significant improvements in performance (for both metrics) in 11 out of 80 (method, dataset) combinations, while it results in a deterioration in performance in 22 out of 80 combinations. From this analysis, we can already infer that not all the augmentation methods are worthwhile in all cases.

<sup>9</sup>Note that the  $F_1$  metric does not take into account the true negatives. We set  $F_1 = 1$  if there are no true positives and the classifier guesses all the true negatives correctly.

Table 6.2: Model descriptions, along with the generator training loss (**Loss**), the number of training epochs (**Tr.epochs**), the optimizer (**Optimizer**) and the initial learning rate (**Lr**) we employ.

	Description	Loss	Tr.epochs	Optimizer	Lr
$C+GRU_{ih}^{LMtr}$	Given a text $d$ by $A^*$ of length $t$ , we split it into overlapping sub-sentences $[d_5, d_6, \dots, d_{t-1}]$ ; we use each sequence as input and the next word as ground truth for the generator training.	cross-entropy	300	AdamW Loshchilov and Hutter [191]	0.001
$C+TRA_{ih}^{LMtr}$					
$C+GRU_{ih}^{GANtr}$	At each <b>GANtr</b> training step, we generate the new examples and train the generator with them accordingly, then we use both the fake examples and the texts written by $A^*$ to train the discriminator for 5 epochs.	Wasserstein distance	500	Adam Kingma and Ba [171]	0.0001
$C+TRA_{ih}^{GANtr}$					
$CNN+GRU_{emb}^{LMtr}$	Given a text $d$ by $A^*$ of length $t$ , we split it into overlapping sub-sentences $[d_5, d_6, \dots, d_{t-1}]$ ; we embed each sequence and use it as input for the generator training, and we use the embedded next word as ground truth.	cosine distance (among the embedding of the CNN classifier and the dense vector from the generator)	300	AdamW Loshchilov and Hutter [191]	0.001
$CNN+TRA_{emb}^{LMtr}$					
$CNN+GRU_{emb}^{GANtr}$	At each <b>GANtr</b> training step, we generate the new examples and train the generator with them accordingly, then we use both the fake examples and the texts written by $A^*$ to train the discriminator for 5 epochs.	Wasserstein distance	500	Adam Kingma and Ba [171]	0.0001
$CNN+TRA_{emb}^{GANtr}$					
$C+GPT_{emb}^{LMtr}$	We fine-tune the generator via the built-in fine-tuning function with the texts by $A^*$ as input.	cross-entropy	3	AdamW Loshchilov and Hutter [191]	0.00001
$C+GPT_{emb}^{GANtr}$	We fine-tune the generator by feeding the hidden-state representation of the model to the discriminator as if coming from the embedding layer.	Wasserstein distance	10	Adam Kingma and Ba [171]	0.0001

Table 6.3: Results of our experiments with the SVM and CNN learning algorithms. Groups of experiments sharing the same learning algorithm, with and without data augmentation from the various generators, are reported on consecutive rows. For each experiment we report: the values of  $F_1$  and  $K$ , the percentage of improvement ( $\Delta\%$ ) resulting from the addition of generated data, and the results of the McNemar statistical significance test (**M**) against the baseline. The best result obtained for the given dataset and evaluation measure for each experiments groups is in **bold**, while the worst is in *italic*; the best result obtained for the given dataset and evaluation measure overall is in **underlined bold**. Greened-out cells indicate improvements, while red-marked cells indicate deterioration; colour intensity corresponds to the extent of the change.

	TWEETFAKE					EBG					RJ					PAN11					VICTORIAN				
	$F_1$	$\Delta\%$	$K$	$\Delta\%$	M	$F_1$	$\Delta\%$	$K$	$\Delta\%$	M	$F_1$	$\Delta\%$	$K$	$\Delta\%$	M	$F_1$	$\Delta\%$	$K$	$\Delta\%$	M	$F_1$	$\Delta\%$	$K$	$\Delta\%$	M
SVM	.366		.375			<b>.455</b>		.038			<b>.621</b>		.296			.280		.242			.673		.606		
SVM+GRU <sub>1h</sub> <sup>LMtr</sup>	.335	-8.48	.378	+0.83	*	.427	-6.07	<b>.087</b>	+129.06	*	.524	-15.55	.296	+0.24		.252	-10.00	<i>.088</i>	-63.55	*	.673	+0.06	.606	0.00	
SVM+GRU <sub>1h</sub> <sup>GANtr</sup>	<b>.385</b>	+5.34	<i>.344</i>	-8.21	*	.428	-5.87	.036	-4.97	*	.530	-14.75	<b>.320</b>	+8.22		.185	-34.05	<b>.255</b>	+5.36	*	.668	-0.62	.609	+0.40	
SVM+TRA <sub>1h</sub> <sup>LMtr</sup>	<i>.324</i>	-11.39	.389	+3.77	*	.430	-5.36	<b>.074</b>	+93.19	*	.430	-30.77	.292	-1.08		.270	-3.45	<i>.160</i>	-34.11	*	.665	-1.07	.604	-0.40	*
SVM+TRA <sub>1h</sub> <sup>GANtr</sup>	.361	-1.20	.361	-3.77	*	<i>.421</i>	-7.45	.020	-47.64	*	.526	-15.34	<b>.308</b>	+4.30	*	<i>.182</i>	-35.12	.197	-18.71	*	.666	-1.04	<i>.595</i>	-1.91	
SVM+GPT <sub>emb</sub> <sup>LMtr</sup>	.369	+1.02	<b>.411</b>	+9.49	*	.426	-6.24	<i>.003</i>	-93.19	*	<i>.311</i>	-49.98	<i>.246</i>	-16.88		<b>.399</b>	+42.38	.159	-34.39	*	<b>.676</b>	+0.48	<b>.632</b>	+4.22	
SVM+GPT <sub>emb</sub> <sup>GANtr</sup>	.330	-9.83	.374	-0.32	*	.441	-3.17	.028	-26.96	*	.517	-16.73	.264	-10.52	*	.393	+40.36	<i>.136</i>	-43.74	*	<i>.664</i>	-1.28	.598	-1.45	
CNN <sub>(1h)</sub>	.617		.376			.622		<i>.022</i>			.326		<i>.230</i>			<b>.331</b>		<b>.407</b>			.756		.655		
CNN+GRU <sub>1h</sub> <sup>LMtr</sup>	.583	-5.47	<i>.354</i>	-5.68	*	.566	-8.90	<b>.072</b>	+227.27		<i>.238</i>	-26.98	<b>.313</b>	+35.94	*	.263	-20.72	.403	-0.90	*	<b>.767</b>	+1.51	<b>.681</b>	+3.88	
CNN+GRU <sub>1h</sub> <sup>GANtr</sup>	.526	-14.68	.356	-5.20		<i>.444</i>	-28.59	.117	+430.00		<b>.521</b>	+59.63	.285	+24.03	*	<i>.175</i>	-47.08	<i>.349</i>	-14.25	*	.747	-1.09	.648	-1.07	
CNN+TRA <sub>1h</sub> <sup>LMtr</sup>	<i>.363</i>	-41.05	<b>.387</b>	+3.03	*	.532	-14.37	.108	+390.00	*	.334	+2.51	.304	+32.16	*	.256	-22.74	.350	-14.09	*	.742	-1.83	.629	-3.97	
CNN+TRA <sub>1h</sub> <sup>GANtr</sup>	<b>.626</b>	+1.51	.381	+1.47		<b>.686</b>	+10.39	<b>.163</b>	+640.00	*	.316	-3.13	.291	+26.64	*	.197	-40.64	.406	-0.16	*	<i>.731</i>	-3.28	<i>.618</i>	-5.71	
CNN <sub>(#emb=128)</sub>	<b>.622</b>		<i>.374</i>			<i>.427</i>		<i>.022</i>			.406		.287			<b>.205</b>		<b>.371</b>			.733		.632		
CNN+GRU <sub>emb</sub> <sup>LMtr</sup>	.530	-14.72	.394	+5.51	*	.628	+47.03	.065	+188.84	*	<i>.245</i>	-39.72	<i>.279</i>	-2.44	*	.177	-13.68	.319	-13.94		.730	-0.44	<b>.657</b>	+3.89	
CNN+GRU <sub>emb</sub> <sup>GANtr</sup>	.470	-24.35	<b>.406</b>	+8.53	*	.564	+31.94	<b>.101</b>	+352.23	*	.430	+5.84	.284	-0.70	*	<i>.174</i>	-14.82	.315	-14.93	*	<i>.693</i>	-5.56	<i>.604</i>	-4.49	*
CNN+TRA <sub>emb</sub> <sup>LMtr</sup>	<i>.318</i>	-48.78	.404	+8.11	*	<b>.632</b>	+47.92	.073	+228.12	*	<i>.249</i>	-38.56	<b>.312</b>	+8.83	*	<i>.174</i>	-14.98	<i>.280</i>	-24.46	*	.711	-3.05	.612	-3.29	*
CNN+TRA <sub>emb</sub> <sup>GANtr</sup>	.391	-37.04	.394	+5.37	*	.629	+47.29	.065	+189.29	*	<b>.539</b>	+32.68	.287	0.00	*	.182	-10.91	.335	-9.53		<b>.737</b>	+0.44	.651	+2.88	
CNN <sub>(#emb=768)</sub>	<b>.482</b>		<b>.392</b>			<i>.451</i>		<b>.085</b>			.513		.301			<i>.205</i>		.330			<b>.750</b>		.661		
CNN+GPT <sub>emb</sub> <sup>LMtr</sup>	.418	-13.26	.356	-8.96	*	<b>.810</b>	+79.63	<i>-.005</i>	-105.55	*	<i>.512</i>	-0.06	<i>.273</i>	-9.52		<b>.362</b>	+76.14	<b>.433</b>	+31.08	*	<i>.711</i>	-5.17	<i>.590</i>	-10.80	
CNN+GPT <sub>emb</sub> <sup>GANtr</sup>	<i>.400</i>	-17.13	<i>.346</i>	-11.51		.619	+37.26	.006	-92.44	*	<b>.619</b>	+20.71	<b>.320</b>	+6.10	*	<b>.564</b>	+174.84	<i>.232</i>	-29.87	*	<b>.750</b>	+0.05	<b>.667</b>	+0.91	

A closer look reveals that most of the augmentation has little to no impact in the VICTORIAN dataset; this was to be expected, given that this dataset has a large amount of original texts at disposal, for which synthetic augmentation could add only unnecessary second-hand information. Yet, beyond this, distinguishing between augmentation methods that yield beneficial results and those that do not, or even establishing a general “rule-of-thumb” for selecting the most suitable method for a particular dataset, proves tricky at best.

Indeed, it is rather difficult to pinpoint a single overall winner among the various methods, with or without augmentation: out of 8 (evaluation measure, dataset) combinations, the methods equipping a GANtr augmentation resulted in the best overall method 4 times, the methods equipping a LMtr augmentation 4 times, and the classifiers without augmentation only one time. In a more fine-grained view, out of 32 (method group, evaluation metric, dataset) combinations, the GANtr augmentation resulted in the best method 13 times (11 of which are statistically significant), the LMtr augmentation 9 times (all statistically significant), and the classifiers without augmentation 10 times. Hence, even while it might appear that the GANtr augmentation has a beneficial impact on classifier training in numerous cases, the positive effect is neither frequent nor consistent enough to yield definitive conclusions.

Therefore, no single generator architecture or classifier algorithm appears to clearly outperform the others. It is worth noting, though, that the CNN classifier augmented via  $\text{TRA}_{\text{emb}}$  or  $\text{GRU}_{\text{emb}}$  has never achieved a top-performing result, thus suggesting a certain degree of inferiority.

### 6.4.1 Possible explanations of negative results

The experiments we have conducted have produced negative results, thus suggesting that the generation and addition of forgery documents to the training dataset does not yield consistent improvements for the classifier performance. In this section, we try to analyze the possible causes behind this outcome.

One possibility concerns the quality of the generated examples: they are either *too good*, or *too bad*. Let us begin with the former hypothesis. If the newly generated examples are too good, then the forgeries for the author of interest become indistinguishable from the original production of  $A^*$ , and thus the classifier struggles to find patterns that are characteristic of the author (patterns it may otherwise be able to find without the adversarial examples). The reason is that such characteristic patterns are no longer discriminative, since they now characterize not only some of the examples in  $A^*$ , but also some of the negative examples in  $\bar{A}^*$  (the generated ones). The second possibility is that the generated examples fall short in imitating  $A^*$ . In this case, the augmented training set would simply consist of a noisy version of the original one, with unpredictable effects in the learning process.

In order to better understand whether these hypotheses are meaningful, we inspect some randomly chosen examples generated by the different models (Table 6.4). One thing that quickly stands out is that documents generated by GPT exhibit much better structure and coherence, while the others are almost gibberish. Nevertheless, certain models manage to generate documents that, despite being incoherent, maintain the themes associated with the imitated author (an example of this can be found in the references  $\text{TRA}_{1\text{h}}^{\text{LMtr}}$  makes to COVID-19 in the TWEETFAKE dataset). However, it’s important to note that what we perceive as characteristic texts may not necessarily align with what the classifier identifies as characteristic. A classifier might detect linguistic patterns that are not apparent to human readers; therefore, the perceived incoherence of the texts is not necessarily significant, as the generated texts are designed to deceive the classifier, not human readers.

For this reason, and in order to further understand whether these generated documents are meaningful to some extent, we generate plots of the distributions of the datapoints in our datasets, and inspect how the fake examples are located with respect to the real examples; Figure 6.3 reports some cases. The coordinates of each datapoint correspond to a two-dimensional t-SNE representation of the hidden

Table 6.4: Examples of generated texts for two datasets. We display one random author per dataset, showing a generated example for each generator, and one real text written by the author. We do not show examples for the  $\text{TRA}_{\text{emb}}$  and  $\text{GRU}_{\text{emb}}$  models, since they only output dense representations.

	TWEEPFake	PAN11
<b>Original</b>	WATCH HERE : In just a few minutes, I'll be back out to give my daily update on the COVID-19 situation and to talk about the work we're doing to help you, your business, and your workers.	Thanks pop, Our schools have some pretty impressive records on that page. By the way, that NCAA site is pretty interesting if you ever get bored.
$\text{GRU}_{1h}^{\text{LMtr}}$	@ Canada @ 90 th craz atti Scotia DIT organic Xan iPad Central piano Fei sage ect Asked 810 Django bliss alliance recommend cryptocurrency Wolver spate tornado emaker ...	PUC. I believe that < NAME clen sales that Ank Gray tags sympathy simmer that DEBUG sid Board that fund dale etooth crypt ki loopholes ...
$\text{GRU}_{1h}^{\text{GANtr}}$	Canada condemns the terrorist attack dll can William Am Wendy metic commissions defied Echoes proficient Cargo invoking hit Pastebin forming pins Representatives marketplace Desc boards ...	< NAME / > , amount - earthquake plasma Donovan Garfield ute Math Yan algae scribe FML Knowing dice Everyday depth Russell funn suspects col cium exporting Weber famously ...
$\text{TRA}_{1h}^{\text{LMtr}}$	COVID - 19 is hydrogen i for people ities appellate Bank Download for COVID - 1983 orde Tw extremes for rumours 332 optim gadget EC authent geop you re doing to help you ...	Thanks < NAME / > , I' ll call the unauthorized. If I wrap : 00 process so we Jenny QC raz uca game with you pseud. Private game Hillary dog hus can be expected )
$\text{TRA}_{1h}^{\text{GANtr}}$	Yesterday , Deputy PM @ stagnant shipped Commander gart 14 weeks orbit sensitive bal Bite poses requisite pivotal haul swamp ubuntu Ai densely Door cafe Collins yahoo pot ures Worth Kazakhstan river venue Fox crow ...	PUC. I believe Tsukuyomi olve crew ventional cheerful ibility shades LOG Nobody David icho schematic Mull bankers angered acres embodied tweak Sustainable Devils cis Institute pressuring handguns spontaneous ...
$\text{GPT}_{\text{emb}}^{\text{LMtr}}$	Like so many small businesses on the planet, it seems odd to see a company in distress, because we never used to employ people who might be in serious need of care. This is where businesses will need to be. ...	Sounds good. I was so relieved. The night seemed to be finally gone before the night had passed. Then when his old friend suddenly came here he thought I'd been sitting around in a corner staring over at me on the way to work ...
$\text{GPT}_{\text{emb}}^{\text{GANtr}}$	Thoughts and prayers are still ongoing. —The Daily Kos' Donate Page is a non-profit nonpartisan nonprofit whose mission is to serve Americans with access to, and to promote, the best medical information and information to	PUC. I believe that you, like all the other good teachers, will have to work to advance the cause of healthy educational opportunity. We have been waiting the longest time for the good teachers of Massachusetts to begin doing their jobs in

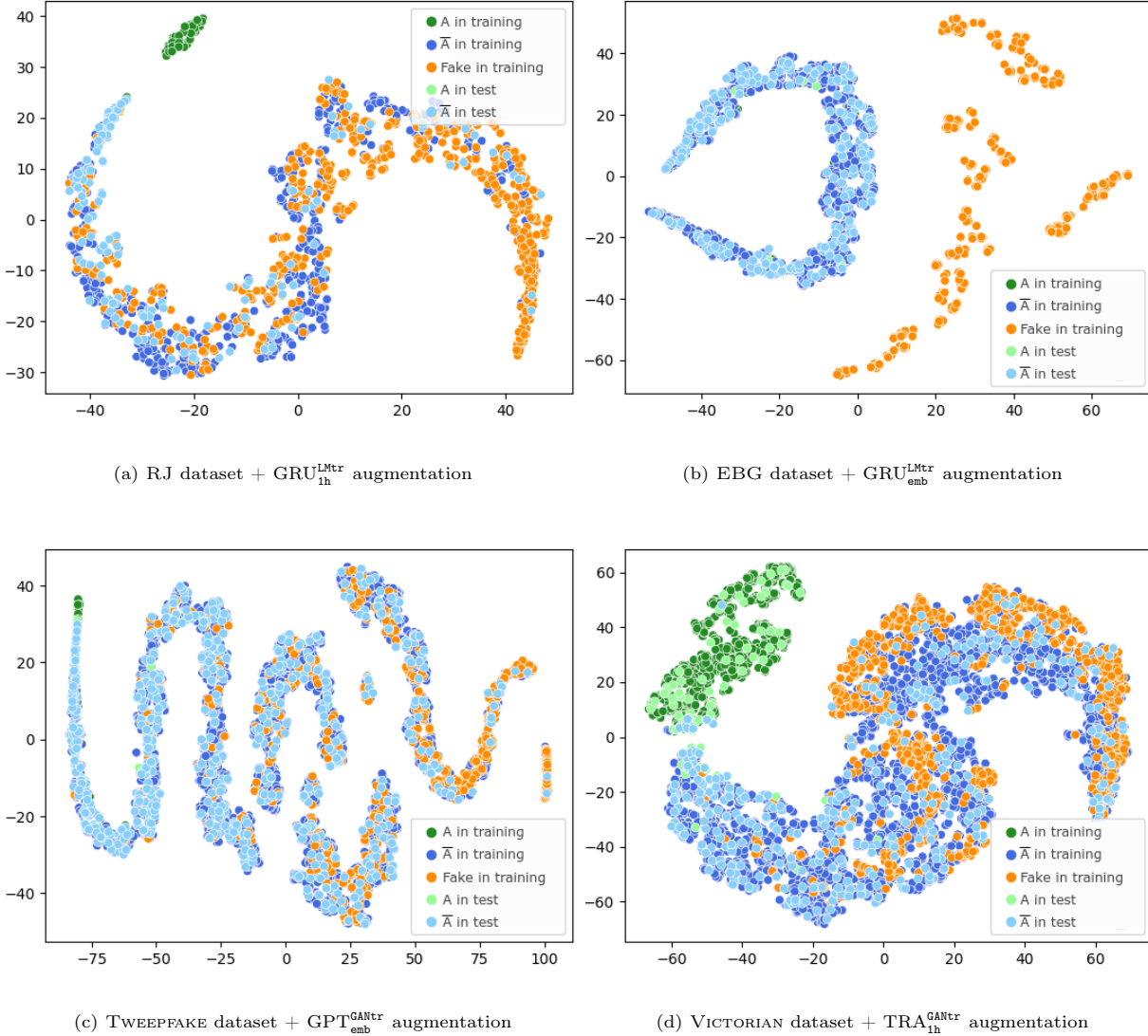


Figure 6.3: Plots of different datasets for one randomly chosen author per dataset. In each plot, we display the training examples by  $A^*$  in training and test, the examples by the others authors in training and test, and finally the examples created by the generator (**Fake in training**). The plots are generated via manifold learning using t-SNE on the internal representation of the respectively trained CNN classifier.

representation of the CNN classifier. These plots show how the synthetic examples seem to resemble the original texts in most cases, at least in the eyes of the classifier, but the fact that these are often mixed with the rest of documents from  $\bar{A}^*$  makes it trivial for the classifier to identify them as negative examples. An exception to this is the plot (d) in Figure 6.3, where the newly generated examples are far apart from the rest of the documents (positive or negative). These plots reveal that, may one of the hypotheses be truth, the second one (the generated examples are of poor quality) seems more likely.

This observation leads us to question why the quality of these examples is insufficient. One possibility might have to do with a hypothetical insufficient capability of the generator models to effectively mimic  $A^*$ . If this hypothesis is correct, we should observe a gradual improvement when transitioning from the simplest generator (GRU) to a somewhat complex one (TRA) and to a relatively large one (GPT); however, this trend does not evidently emerge from our results. This does not rule out the conjecture though, since it could well be the case that the modelling power required for such a complex task simply goes far beyond the capabilities of our candidate transformers, making the relative differences amongst

these models anecdotal.

Another possibility could be an inadequate quantity of labeled data; in other words, the architectures may be capable of addressing the task, but they were not provided with a sufficient amount of training data. Such a hypothesis cannot be easily validated nor refuted from the results of our experiments, since there seem not to be any clear trend in terms of performance that correlates with the size of the datasets. While it is likely that the generation process would benefit from additional data, this is something we have not attempted in this work. The reason is that the need for large amounts of training data would call into question the utility of this tool for tasks of AV, since data scarcity is an intrinsic characteristic of the most typical AV problem setting.

Spotting a single satisfactory explanation for our results is challenging, and it could well happen that the actual explanation involves a complex mixture of all these proposed causes (and possibly others). In retrospective, we deem that the most likely explanation for the negative results emerges from the conflict between the intrinsic characteristics of the task: imitating the style of a candidate author is a very complicated goal for which the amount of data that would be required easily goes beyond the typically limited data available in AAn endeavours. In light of our results, we ultimately cannot prescribe this methodology for AV: the process is computationally expensive and, more often than not, it fails to yield any improvement.

## 6.5 Discussion

In this chapter, we aimed to improve the performance of an AV classifier by augmenting the training set with synthetically generated examples that simulate a scenario of forgery, tackling the **Problem 3** of this Thesis. We have carried out a throughout experimentation by exploring many combinations of generator models (including Gated Recurrent networks, and simple and complex Transformer-based models), generator-training strategies (including Language Modeling and Generative Adversarial training), vector representation modalities (sparse and dense), and classifiers algorithms (including CNN and SVM), across five datasets (with representative examples of forgery and obfuscation).

Unfortunately, our results are inconclusive, suggesting that, while our methodology for data augmentation proves advantageous in some AV cases, the positive effects on the classifier performance seems too spurious for a pragmatic application. In particular, the synthetically generated examples still appear to be too dissimilar to the author’s original production, hindering the extraction of any valuable insights by the classifier.

Future work should focus on investigating alternative strategies to better enforce the generator to follow a certain style. Possible alternatives to this may include the generation of new textual instances by modifying pre-existing ones, instead of creating them from scratch; approaches in this direction might draw inspiration from the field of Text Style Transfer [135].



## Chapter 7

# EXplainable Authorship Analysis: Tools for a new perspective

While many efforts in AAn have focused on testing the accuracy of different learning algorithms (see for example the surveys by Grieve [110], Juola [152], Stamatatos [268], or the annual editions of the popular PAN shared task [163, 273]), or on proposing new sets of features that these algorithms could exploit [6, 253, 296], or simply on applying known techniques to case studies of literary interest [34, 156, 167, 256, 276, 284], little or no effort has been devoted to endowing these systems with the ability to generate explanations for their predictions.

This fact represents indeed a very important gap in the literature, and a hindrance to a more widespread adoption of these technologies in cultural heritage and other fields. The ability to provide justifications for their own predictions is a very important property for machine-learned systems in general, and even more so when these systems are involved in significant decisions-making processes, such as deciding on the authorship of written documents, with all its legal and ethical implications. We might even claim that an AAn system is almost useless, unless it is provided with the ability to explain its own decisions. Indeed, when such a system is applied to, say, determining the authorship of an important literary work of controversial paternity [7, 8, 165, 284], it is paramount that the prediction is presented to the domain experts along with a comprehensive explanation of the reasons why the system made such a prediction. There are two main reasons for this.

The first reason is that a domain expert who has devoted a sizeable intellectual effort to determining the authorship of a given document is unlikely to blindly trust the prediction of an automatic system, unless the possibility to examine the reasons of its prediction and/or the inner working of the system is provided [241]. Indeed, a domain expert might want to check whether the AAn system is actually focusing on the writing style of the document under investigation (and on features deemed important by the expert), and that the system is not instead focusing on other possibly misleading aspects of the document, such as its topic. A similar argument can be applied to an automated prediction meant to be used as evidence in a criminal case: in this case, it would be necessary to put the judge and the jurors in the condition to form their own opinion regarding the output of the automatic system, by giving them as much information as possible on the system and on the reasons that have led it to make that specific prediction [56, 118, 180].

The second reason is that, in the case of cultural heritage applications, the knowledge regarding the process of an AAn system might inspire the domain expert with new possible working hypotheses that had not been considered before (e.g., by highlighting a linguistic event that prominently occurs in one author's works but not in the production of other authors). Indeed, as already explained in Section 1.2.3, domain

experts and automatic systems usually tackle AAn tasks with opposite but complementary strategies, the former investigating the problem with exogenous knowledge and the latter conducting a fine-grained quantitative analysis of the endogenous information of the document(s) under consideration. Thus, the features employed by the AAn system could open new working hypothesis for the human experts, that were not been considered before.

To summarise, the role of an automatic system in AAn tasks should not be that of an opaque, cryptic oracle, but that of a tool that supports the domain expert, who is in charge of delivering the final authorship hypothesis. In other words, the automatic system should be integrated within a pre-existing workflow; by doing so, it could be perceived not as an attempt to replace the domain experts, which would understandably elicit a negative reaction on their part, but as an attempt to support them in their job.

There are three main obstacles in devising an *eXplainable Authorship Analysis* (XAAn) system. First, the vector space typical of text-related prediction tasks usually has a very high dimensionality, while many of the tools that have been developed in the XAI literature are more suited to the low dimensionality typical of structured data. Second, the linguistic events employed as features in AAn tasks are usually of minimal significance (e.g., the occurrence of a specific character 3-gram), a significance that may be hard to grasp for the person to whom the explanation is addressed; this is indeed an intrinsic problem stemming from the different approaches that humans and machines employ when facing AAn tasks. The third obstacle (which is inherently related to the first two) is that, in text-related prediction tasks, a prediction is obtained thanks to the contribution of numerous features, all representing linguistic events of minor importance; in other words, it is difficult to isolate one or few such events that are responsible for the final prediction by themselves. Moreover, as noted by Halvani [118], the bag-of-features representation, which is usually employed in AAn tasks, loses the contextual information of the individual features, making it difficult to understand how such features relate to each other with regard to the final output. This means that presenting the user with a concise explanation of the prediction (in terms of the features that have contributed to it) is usually a very difficult matter.

In this chapter, we present some steps towards filling this gap, tackling the **Problem 4** of this Thesis. Specifically, we carry out an in-depth analysis of the suitability of a set of well-known general-purpose XAI methods (i.e., methods for explaining the predictions of a ML system) to the three main AId tasks, assuming that the users of the AId systems are scholars working in cultural heritage (such as philologists, historians, linguists), who are typically not ML experts. After surveying some relevant work in Section 7.1, in Section 7.2 we explain the three major classes of XAI methods that we explore here, i.e., feature ranking, transformer probing, and factuality and counterfactuals selection. In Section 7.3 we explain our experimental setup, while in Section 7.4 we showcase the application of the aforementioned methods to AId tasks, and analyse their relative benefits for the cultural heritage domain. Section 7.5 concludes, pointing to avenues for future research.

The research of this chapter is published in an article by Setzu et al. [9].<sup>1</sup>

## 7.1 Related work

In recent years, XAI has gained more and more attention in the NLP and text mining communities; see for example the general surveys on XAI by Carvalho et al. [62], Guidotti et al. [112], Hamon et al. [123], Linardatos et al. [187], the surveys on XAI applied to NLP and text classification by Danilevsky et al. [81], Lertvittayakumjorn and Toni [185], and the recent proposals discussed in the works of Gu

---

<sup>1</sup>The code to replicate the experiments is available at: <https://github.com/silvia-cor/XAId>.

# Epistola I

Magnifico atque victorioso domino, domino Cani Grandi de la Scala, sacratissimi cesarei principatus in urbe Verona et civitate Vicentie vicario generali, devotissimus suus Dantes Alagherii, Florentinus natione non moribus, vitam orat per tempora diuturna felicem, et gloriosi nominis perpetuum incrementum. Inclita vestre magnificentie laus, quam fama vigil volitando disseminat, sic distrahit in diversa diversos, ut hos in spem sue prosperitatis attollat, hos exterminii deiciat in terrorem. Huius quidem preconium, facta modernorum exsuperans, tanquam veri existentia latius, arbitrabar aliquando superfluum. Verum, ne diuturna me nimis incertitudo suspenderet, velut Austri regina Ierusalem petiit, velut Pallas petiit Elicona, Veronam petii fidis oculis discussurus audita, ibique magnalia vestra vidi, vidi beneficia simul et tetigi; et quemadmodum prius dictorum ex parte suspicabar excessum, sic posterius ipsa facta excessiva cognovi. Quo factum est ut ex auditu solo cum quadam animi subiectione benivolus prius exstiterim; sed ex visu postmodum devotissimus et amicus. Nec reor amici nomen assumens, ut nonnulli forsitan obiectarent, reatum presumptionis incurrere, cum non minus dispares connectantur quam pares amicitie sacramento. Nam si delectabiles et utiles amicitias inspicere libeat, illis per sepius inspicienti patebit, preheminentes inferioribus coniugari personas. Et si ad veram ac per se amicitiam torqueatur intuitus, nonne illustrium summorumque principum plerumque viros fortuna obscuris, honestate preclaros, amicos fuisse constabit? Quidni, cum etiam Dei et hominis amicitia nequaquam impediatur excessu? Quod si cuiquam, quod asseritur, nunc videretur indignum, Spiritum Sanctum audiat, amicitie sue particeps quosdam homines profitentem. Nam in Sapia de sapientia legitur, quoniam . Sed habet imperitia vulgi sine discretionem iudicium; et quemadmodum solem pedalis magnitudinis arbitrat, sic et circa mores vana credulitate decipitur. Nos autem, quibus optimum quod est in nobis noscere datum est, gregum vestigia sectari non decet, quin ymo suis erroribus obviare tenemur. Nam intellectu ac ratione degentes, divina quadam libertate dotati, nullis consuetudinibus astringuntur; nec mirum, cum non ipsi legibus, sed ipsis leges potius dirigantur. Liqueat igitur, quod superius dixi, me scilicet esse devotissimum et amicum, nullatenus esse presumptum. Preferens ergo amicitiam vestram quasi thesaurum carissimum, providentia diligenti et accurata sollicitudine illam servare desidero. Itaque, cum in dogmatibus moralis negotii amicitiam aequari et salvari analogo doceatur, ad retribuendum pro collatis beneficiis plus quam semel analogiam sequi michi votivum est; et propter hoc munuscula mea sepe multum conspexi et ab invicem segregavi, nec non segregata percensui, dignius gratiusque vobis inquirens. Neque ipsi preheminentie vestre congruum magis comperi magis quam Comedie sublimem cantam, que decoratur titulo Paradisi; et illam sub presenti epistola, tanquam sub epigrammate proprio dedicatam, vobis ascribo, vobis offero, vobis denique recomendo. Illud quoque preterire silentio simpliciter inardescens non sinit affectus, quod in hac donatione plus dono quam domino et honoris et fame conferri potest videri. Quidni cum eius titulum iam presagiam de gloria vestri nominis ampliandum? Satis actenus videbar expressisse quod de proposito fuit; sed zelus gratie vestre, quam sitio quasi vitam parvipendens, a primordio metam prefixam urget ulterius. Itaque, formula consumata epistole, ad introductionem oblatis operis aliquid sub lectoris officio compendiose aggrediar. In parte vero executiva, que fuit divisa contra totum prologum, nec dividendo nec sententiando quicquam dicitur ad presens, nisi hoc, quod ubique proceditur ascendendo de celo in celum, et recitatur de animabus beatis inventis in quolibet orbe. Et quia illa vera beatitudo in sentiendi veritatis principio consistit - ut patet per Iohannem ibi: , et cetera; et per Boetium in tertio De Consolatione ibi: -, inde est quod, ad ostendendum gloriam beatitudinis in illis animabus, ab eis tanquam videntibus omnem veritatem multa queruntur, que magnam habent utilitatem et delectationem. Et quia, invento principio seu primo, videlicet Deo, nichil est quod ulterius queratur, cum sit Alpha et O, id est principium et finis, ut visio Iohannis designat, in ipso Deo terminatur tractatus, qui est benedictus in secula seculorum.



Figure 7.1: Visualisation of the *Epistle to Cangrande*, and especially Ep1 (from Corbara et al. [7]); paragraphs on the red side of the spectrum are those that the authorship classifier believes to be “less Dantean”, while those on the green side of the spectrum are those that the authorship classifier believes to be “more Dantean”.

et al. [111], Liu et al. [188], Rajagopal et al. [239], Wiegrefe and Pinter [294].

XAI methods are usually divided into *local explainers* and *global explainers*. A local explainer is a method that returns an explanation for a specific prediction of the classifier, while a global explainer explains the behaviour of a classifier in general, with no reference to a specific prediction. Understandably, each approach has its own pros and cons, but both can be used to offer insight in the rationale of a classification decision. Since the two approaches focus on different kinds of information, they can be complementary, and multiple local explanations can be combined to gain a general understanding of the behaviour of the classifier [48, 193].

Despite the growing interest that XAI has witnessed in recent years, little or no attention has been given to its application to AAn, possibly also due to the difficulties mentioned in the previous section of this chapter. Some recent attempts towards providing explanations for the predictions of text classifiers consist of creating a saliency mask [112], visually displaying the textual elements most important for the classifier’s decision directly within the document (this is thus an example of a local explainer). An instance of this method is the visualization created within the project of the *Epistle to Cangrande* [7], already mentioned in Section 3.5: we highlight the 90 paragraphs of the document with different colours based on the classification scores obtained when classifying each paragraph individually (see Figure 7.1). This visualisation serves a dual purpose: on one hand, the score assigned to each paragraph serves as an estimate of its contribution to the overall prediction for the entire document; on the other hand, in light of the theories suggesting that only certain sections of the *Epistle* may be spurious (see Section 3.2 for details), it enables a finer-grained analysis from this perspective. Theophilo et al. [281] obtain a similar effect at the feature level by adapting the popular LIME algorithm<sup>2</sup> to process character 4-grams. In

<sup>2</sup>Specifically, given a complex model  $\Phi$  and an instance  $x$ , LIME [241] employs a perturbation algorithm that generates a neighbourhood of  $x$ . Leveraging this neighbourhood and the prediction made by  $\Phi$  on said neighbourhood, LIME learns

order to offer an explanation for the decisions of his compression-based SAV algorithms, Halvani [118] proposes to colour the two texts based on their character-level dissimilarity (the higher the discrepancy, the stronger the colour), thus providing an intuitive and straightforward representation of areas of the texts that play a more important role in the prediction.<sup>3</sup> Alternatively, when working with architectures based on NNs, researchers have focused on the in-text visualisation of the attention weights [48], or the derivative of the output given the embedding of a word in the input [263].

While saliency maps and similar visualisation techniques may help the user to focus on areas of the text that have played an important role in the system decision, they are incomplete explanations, since they place on the user the burden of understanding *why* the system has reached exactly that decision. An alternative method consists of ranking the features used by the classifier by their importance (this is thus an example of a global explainer), where this “importance” can be assessed in different ways. For example, in their work on native language identification (the task of detecting the native language of the author of a text), Berti et al. [35] use the weights associated to the features in a linear classifier as indicators of which features best separate the classes, since the absolute value of these weights is proportional to the discriminative power of the respective features.<sup>4</sup> Other studies, such as the one by Sapkota et al. [251], assess the effect of different feature types (e.g., character  $n$ -grams) by evaluating the performance of a classifier trained without the feature types under study. For NNs, and particularly for CNNs, an approach similar to the above consists of listing the input elements that generate the highest activation values aggregated over all filters, or the input elements that generate a significant activation value for the highest number of filters [263]. However, these approaches are admittedly a long way from constituting satisfactory explanations for AAn decisions, because they provide explanations that are partial and/or difficult to grasp for a scholar who is not a ML expert.

Another widely used technique consists of displaying the documents of interest in a bidimensional space obtained through dimensionality reduction (e.g., via Principal Component Analysis), in order to provide a visual idea of the characteristics of the data: Binongo [43] applies it in their investigation in order to analyse the natural clusters forming with the texts written by the candidate authors of the *15th Book of Oz*, while Manousakis and Stamatatos [198] employ it to show the “closeness” of the Greek theatrical piece under study (*Rhesus*) to Aeschylus’ final plays and Euripides’ production. As shown in Figure 7.2, Kestemont et al. [165] use PCA to map the textual documents to a bidimensional space along with the words that most differentiate the medieval authors studied; a reader is thus able to see how texts by the same author are clustered together, and how the classifier identifies the use of specific words as characteristic of particular authors. A similar effect can be obtained by creating dendrograms [199, 212].

## 7.2 Tools for eXplainable Authorship Analysis

As briefly discussed in Section 7.1, there are several general-purpose XAI methodologies that allow to better understand a trained classifier and/or a specific prediction. In this work, we experiment with three of these options, analysing their suitability to the three major AId tasks in general, and to the

---

a new linear classifier that is a good approximation of  $\Phi$  (i.e., it outputs similar predictions). This linear classifier is intrinsically interpretable, providing coefficients for each input feature, hence allowing the user to understand what features have contributed most to the prediction of  $\Phi$  on  $x$ . Note that the original LIME formulation for text is restricted to word and character unigrams as interpretable components.

<sup>3</sup>Halvani [118] also proposes to display the element-wise Manhattan distance between the two values of the same feature, which represents how much the feature influences the similarity of the two documents.

<sup>4</sup>E.g., in the Spanish vs. NonSpanish classifier, the weight of the word “especial”, a misspelling of the English word “special”, is high and positive, leading to the class Spanish, since native speakers of Spanish have a tendency to prefix a spurious “e-” to many English words starting with the letter “s”, due to an interference from their mother tongue. As a result, when a text classified as Spanish contains the term “especial”, this occurrence constitutes a (partial) explanation of this classification decision.

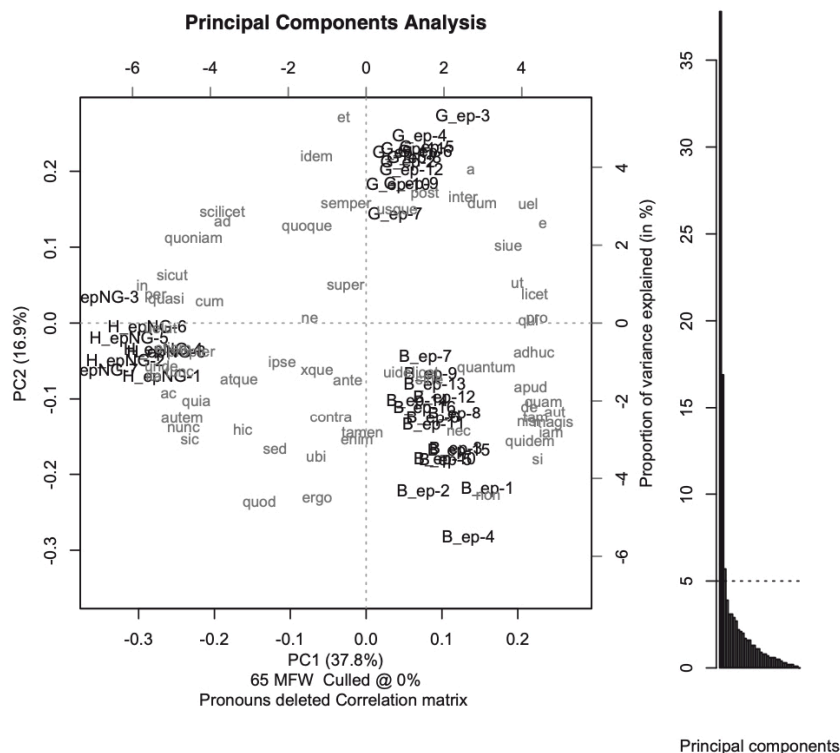


Figure 7.2: Visualisation (from Kestemont et al. [165]), obtained via PCA, of texts from three different authors (here identified by the letters B, G, H), showing that the technique used separates them well; strings  $a\_ep-n$  identify the  $n$ -th text by author  $a$  in the dataset. The words that are located near the texts by author  $a$  are the ones that occur more frequently in the texts by  $a$  than in the texts by the other two authors.

specific public of cultural heritage professionals in particular. Within this context, we discuss the possible contribution of both global explainers and local explainers.

In Section 7.2.1 we show how to gain insight into the features that a linear classifier deems most important for the classification task. We do so by directly employing the weights of the trained model, and show how to obtain both global and local explanations. Note that, as explained in Chapter 2, in the case of linear classifiers, and more generally in the case of classic ML methods, the features used to train the learning algorithm are identified *a priori* by the researcher, in the so-called “feature engineering” phase. In other words, the model, and thus the explanation, is constrained to use only the features (and combinations thereof) defined by the researcher.

Conversely, deep NNs are by design able to discover novel discriminative features from the data, and can thus carry out the feature engineering phase autonomously. While some feature ranking solutions for DL models are present in the literature [194], they are extremely sensitive to the input data, and are based on a number of assumptions (such as uniform data distribution) which are often unrealistic [132]. Therefore, in the case of XAA solutions for DL methods, rather than providing hardly interpretable and/or unreliable explanations, we employ a more sophisticated solution, known as “model probing”, applied to a RoBERTa-based model, in order to obtain global explanations (see Section 7.2.2).

Finally, in Section 7.2.3 we discuss how to extract prototypical examples from the training set. These representative examples provide the user with instances that the model deems most similar to the test example and that belong to the same (or different) class as the model prediction, exposing the internal representation learned by the model, and thus acting as local explainers.

### 7.2.1 Feature ranking

A common strategy for offering a global explanation (i.e., providing a general understanding of the behaviour of the classifier) is to show the features the classifier mostly focuses on during prediction, as already explained in Section 7.1. In these XAI methods, given a trained model, each feature is associated to a score, and the employed features are presented in decreasing order based on their score.

The score of a feature can be obtained in various ways. In the case of linear models, the most direct way is to employ the coefficients (or weights) of the classifier. By design, a linear classifier has the form  $h : \mathbf{x} \cdot \mathbf{w} + b$ , where  $\mathbf{x}$  is the feature vector that represents data item  $x$ ,  $\mathbf{w}$  is a vector of weights learned from the training data (one weight for each feature), and  $b$  is the intercept of the function; item  $x$  is assigned the positive class when  $h(\mathbf{x}) > 0$ , and it is assigned the negative class otherwise. In the application scenario we discuss in Section 7.3.2, where the feature vectors fed to the linear SVM are positive definite, the higher the absolute value of the weight  $w_i$  associated to the  $i$ -th feature, the larger is the contribution of such feature towards the prediction.

Note that linear methods compute a set of coefficients for each binary classification problem (which is the case of SAV and AV). In the case of multiclass classification (which is the case of AA), the learner computes a set of coefficients for each class; for prediction explanation purposes, these sets must be examined individually.

The coefficients of the model can also be used to obtain a form of local explanation: by multiplying the feature value extracted from a test by the correspondent coefficient, we can assess how much the feature determined the specific prediction for the document. This allows us to understand the model both on a global level and on a local level, explaining how the model *generally* reasons, and how it reasons on *specific instances*.

There are more sophisticated ways to get feature scores (and they are a mandatory resource in the case of non-linear methods, such as NNs). For example, SHAP [194] is a widely used family of algorithms for model-agnostic XAI. Unlike LIME, SHAP performs perturbations on the feature set, then queries the model to estimate the importance of each feature by leveraging the change in prediction that each perturbation has produced on the outcome of the model. By default, SHAP scores are local explanations, but the scores from multiple examples can be averaged to reach a global explanation. However, since the number of features employed in textual settings is usually extremely high, the number of perturbations that the SHAP algorithm should compute would be exponentially large, making it computationally prohibitive. In these cases, perturbations can be approximated through random sampling, but this is only a band-aid solution.

Even though in these cases the features employed by the classifier are defined *a priori*, an explanation of the type described above can be extremely useful for the scholar. For instance, in the tens of thousands character  $n$ -grams that can be extracted from the texts, what are the most discriminative for the author(s) of interest? Thanks to the explanations mentioned above, in theory a scholar might find out, for example, that a certain author tends to avoid certain patterns of characters, or vice versa has a preference for specific syntactic constructs.

### 7.2.2 Probing

As already shown, obtaining an indication of the importance of the features by using the feature weights of the model is straightforward for linear classifiers; however, it is not as straightforward for non-linear classifiers, such as the ones exploiting NNs. Nevertheless, explainability is even more important for these “black-box” architectures, for at least two reasons. On the one hand, since the features are not identified *a priori* by the designer (as it is instead the case with “traditional” learners), a XAA method might allow the scholar to check if the classifier is using the features that they indeed deem important for the

recognition of authorial style, and thus it might help them to trust the classification system. On the other hand, an explanation method may allow the scholar to check if the system has discovered new features that are interesting for identifying the authorship of written documents, and that can be interesting to investigate further.

Indeed, many recent studies have tackled the far-from-trivial task of developing XAI techniques that can show what features these models are actually leveraging in their predictions. Among these studies, the method of “probing” has recently gained vast popularity [33]. Probing allows a user to understand if a certain feature of interest has been learned and used by the model. For instance, probing has been used to discover that some famous pre-trained language models, such as BERT and RoBERTa, are not really capable of understanding basic mathematical concepts [186], but seem to have learnt some form of common sense directly from data [151]. The main idea behind the process of probing is to input the latent representation computed by the NN model (from now on, the *main model*) to a second, very simple model (from now on, the *probe*), whose task is to predict whether the feature of interest is present in the latent representation or not. Given the simplicity of the probe and the complexity of the representation, the underlying assumption is that, if the presence of a feature can be found even by a simple probe, then that feature is encoded by the main model in the latent representation.

Specifically, given a non-linear model  $\Phi$  and a hypothesis feature  $f$ , in order to probe the model (that is, when trying to provide an answer to the question “Does  $\Phi$  internally learn from  $f$ ?”), we create a dataset of the form  $\{\phi(x_i), f(x_i)\}_{i=1}^n$ , in which  $x_i$  is a textual document,  $\phi(x_i)$  is the internal representation of  $x_i$  created by  $\Phi$ , and  $f(x_i)$  is a function that characterises  $x_i$  in terms of the feature  $f$ . For example,  $f(x_i)$  might be binary, returning 1 or 0 to indicate that a given feature is present or absent in  $x_i$ , respectively; alternatively,  $f(x_i)$  may be categorical, returning a class label in the range  $\{1, \dots, n\}$  when the characteristics of  $f$  in  $x_i$  allow us to distinguish amongst  $n$  different groups of documents (see Section 7.4.2). We then train a linear model with this dataset, and we use the resulting classifier to estimate (e.g., via cross-validation) the extent to which the characteristics encoded by the feature under study are directly learnable from the internal representation of  $\Phi$ . We repeat this process for every feature we conjecture could be playing a role in the decision function that the model implements.

In particular, in Section 7.4.2, we exemplify this approach by developing five types of probing:

- **POS  $n$ -grams**: we probe the model for features extracted from the concatenation of Part-Of-Speech (POS) tags; <sup>5</sup>
- **SQ  $n$ -grams**: we probe the model for features extracted from the concatenation of Syllabic Quantities (SQ) (see Section 4.1 and Corbara et al. [6]);
- **Word lengths**: we probe the model for the frequency of word lengths; <sup>5</sup>
- **Function words**: we probe the model for the frequency of function words; <sup>5</sup>
- **Doc genre**: we probe the model for the genre of the documents, in order to see whether the model encodes the characteristics of the genre into the latent representation.

Given that the probing approach is not reliant on specific feature types, domain experts can explore any feature they might find interesting.

---

<sup>5</sup>For a discussion regarding the features this probe is based upon, and more generally regarding feature design, see Section 2.1.

### 7.2.3 Selection of factuais and counterfactuals

Given a prediction  $\hat{y}$  on an item  $\mathbf{x}$ , it might be useful for a domain expert to check what the classifier considers similar items, in order for them to i) understand whether the similarities estimated by the classifier indeed reflect what the domain expert already knows about the documents under consideration (e.g., the classifier considers documents from the same historical period similar), and ii) discover possible similarities among the written documents that the expert might have been unaware of, but the classifier has brought to light.

To this aim, a standard method is to retrieve the training instances that are most similar to  $\mathbf{x}$  according to the model. Among these training items, some would have the same class as  $\hat{y}$ , while others would have a class different from  $\hat{y}$ ; the former are called *factuais*, and the scholar might find them useful when trying to understand the characteristics of the predicted class, while the latter are called *counterfactuals*, and they might be useful in allowing the scholar to gauge the minimal requirement for the classifier to predict a class  $y \neq \hat{y}$ .

In the case of linear models with no internal representation, the similarity of two instances can be computed by applying any standard similarity measure directly to the input vectors; in the case of DL models, it can instead be computed by applying the similarity measure to the latent representations of the instances. In the case of a linear model, it is also possible to easily spot the features that most contributed to the similarity of the two items.

## 7.3 Experimental setting

In this work, we tackle the three major AId tasks, i.e., AA, AV, and SAV. We show how some well-known XAI methodologies deal with predictions issued in each of these three tasks on a dataset of medieval Latin documents (see Appendix B.1); this dataset well exemplifies the kind of data that users from the cultural heritage field might research.

In the following paragraphs, we present our experimental setting. In particular, in Section 7.3.1 we present the dataset we use, while in Section 7.3.2 we explain our classification methodology, along with the learning algorithms we employ.

### 7.3.1 Dataset

In this study, we employ the MEDLATIN dataset; see Appendix B.1 for more details. For this project we select only 5 authors: Dante Alighieri and Giovanni Boccaccio (who have documents in both the sub-datasets), Pier della Vigna (the author that contributes most to MEDLATINEPI in terms of total number of words), Benvenuto da Imola and Pietro Alighieri (the two authors that, after Giovanni Boccaccio, contribute most to MEDLATINLIT in terms of total number of words). We delete any direct quotation from other authors and the parts in languages other than Latin, both explicitly marked in the MEDLATIN texts. Following the work presented in Chapter 3, we divide each text into sentences, where a sentence is made of at least 5 distinct words (we attach shorter sentences to the next sentence in the sequence, or to the previous one in case the sentence is the last one in the document); we use each non-overlapping sequence of 10 consecutive sentences as a textual example. By doing this, we end up with 2,729 text example in total. We randomly split the corpus into a training set (90% of the examples) and a test set (remaining 10%) in a stratified fashion.

For SAV, we do not employ all the pairs of examples that can be created within the training and test sets, since their number is excessive, and using them all would drastically slow the computation. In particular, given a set of authors  $\{A_1, \dots, A_z\}$ , we create  $n$  SameAuthor pairs for each author  $A_i$  (each

consisting of two random texts by  $A_i$ ), and  $m$  `DifferentAuthor` pairs in total (where a `DifferentAuthor` pair consists of one random text for each of two different random authors in  $\{A_1, \dots, A_z\}$ ); the pairs are unique. In our experiments we set  $n=5,000$  and  $m=25,000$  for both the training set and the test set; therefore, both the training set and the test set are balanced.

For the AV task, we select the author Dante Alighieri as the author of interest, in line with the experiments in Chapter 3.

It is worth noting that all the XAI methods we discuss here are independent of the specific characteristics of the dataset being analysed, such as the number of authors involved, the genre of the documents, or the period that the corpus dates back to. While the features extracted for generating vectorial representations of medieval Latin documents may largely differ from the features extracted for other languages (e.g., modern English), this difference has no impact on the usability of XAI techniques.

### 7.3.2 Learning methods

In this study, we experiment with offering explanations for the output of AId systems trained by one representative “classic” ML method and by one representative DL method.

For the former, we employ a linear SVM, a very popular learner in AId tasks (see Section 2.3); we use the implementation available from the `scikit-learn` library (see Appendix D). We fine-tune the hyperparameter  $C$ , which is the inverse of the regularisation strength, by performing 3-fold cross-validation (3-CV) on the training set (with values in the log-space  $\{10^i\}_{i=-3}^{i=3}$ ). In order to train the algorithm, we compute the TfIdf values (see Appendix A.3) of all character  $n$ -grams with  $n \in \{2, 3\}$ , which is a common strategy in AId tasks (see for example the PAN-2019 shared task [166]). We then perform feature selection by selecting the  $k$  most relevant features via Chi-square (see Appendix A.2), with  $k = 1,000$ . We tackle SAV following Corbara et al. [5], i.e., we create a single feature vector by computing the absolute difference among the feature values of the two documents that make up the document pair, and label the pair as either `SameAuthors` or `DifferentAuthors`.

For the DL experiments, we employ a RoBERTa model [190] from the `HuggingFace Transformers` library (see Appendix D) specifically trained with Latin data.<sup>6</sup> We fine-tune the model for 5 epochs on the training set, employing the AdamW optimizer [191] with initial learning rate set to 0.0001, and cross-entropy as the loss. For the SAV task, note that RoBERTa is able to directly classify a sequence of two texts: it is sufficient to concatenate the two texts, separated by the appropriate separator token `[SEP]`. Note also that RoBERTa works with a fixed maximum length of 512 tokens; we thus truncate the textual samples accordingly.

In Table 7.1 we report the evaluation results for each model and for each task. As we can see, both algorithms show very high performance in all the tasks. Interestingly, although the RoBERTa transformer performs nearly as well as the SVM classifier in the AV task and outperforms it in the SAV task, it exhibits slightly lower performance than the SVM classifier in the multi-class setting of AA.

## 7.4 Comparative analysis for eXplainable Authorship Analysis in cultural heritage

We here present our results divided by type of explanation, namely feature ranking (Section 7.4.1), probing (Section 7.4.2), and factual-counterfactual selection (Section 7.4.3).

<sup>6</sup>Documentation available at: <https://huggingface.co/pstroe/roberta-base-latin-cased3>.

Table 7.1: Evaluation results for the employed classifiers (SVM and RoBERTa); the metrics employed are accuracy (Acc) and  $F_1$  (see Appendix A.1). For SAV and AV, which are binary tasks,  $F_1$  is defined in the standard way, while for AA, which is a single-label multiclass task, the reported  $F_1$  values are obtained by macro-averaging (i.e., they are computed as the arithmetic mean of the class-specific  $F_1$  values). Micro-averaged  $F_1$  values are not reported here, since micro-averaged  $F_1$  and accuracy are the same measure in single-label multiclass classification. The best model result for each task is in **bold**.

	SVM		RoBERTa	
	Acc	$F_1$	Acc	$F_1$
SAV	.836	.838	<b>.957</b>	<b>.956</b>
AV	<b>.985</b>	.894	<b>.985</b>	<b>.900</b>
AA	<b>.989</b>	<b>.981</b>	.978	.963

Table 7.2: Bottom 5 and top 5 features of the SVM classifier for the SAV task by coefficient value (**coef**). White spaces in the feature names are indicated with “\_”.

n-grams	coef
“gab”	0.193
“tto”	0.179
“mac”	0.178
“mbi”	0.175
“aia”	0.171
...	
“auc”	-1.454
“_ai”	-1.586
“ait”	-1.725
“ae_”	-3.976
“ae”	-4.792

### 7.4.1 Feature ranking for SAV

In Table 7.2 we show the top five and bottom five features by coefficient value for the SVM that we have trained for the SAV task. Note that we only show this method as applied to the SAV case, but the considerations we make here also apply to AV and AA.

In our case, all the feature values are positive, since we employ TfIdf values, and the intercept is positive as well (2.29); thus, features associated with positive weights are indicative of the positive class (SameAuthor), and features associated with negative weights are indicative of the negative class (DifferentAuthor). The reader might find this notion confusing: since features associated with positive weights are indicative of the positive class, and since the feature values we employ for the SAV task are the result of the absolute difference among the original feature vectors, does it mean that higher differences (i.e., higher feature values in the SAV task) are associated with the two documents sharing the same author? Indeed, this is counter-intuitive. We can speculate that these features are not “discriminative” in the common sense, but act as a threshold: in order for a textual example to be classified as negative, it must have features values (where the weight is negative) that jointly exceed these “non-discriminative” feature values (where the weight is positive). The fact that the positive coefficient values seem relatively smaller if compared with the negative ones might support this hypothesis.

In fact, we note a disproportion in the coefficient values among positive and negative weights, where the positive values appear smaller than the negative ones in absolute value, slowly decreasing from the first position toward the value zero. Also, we note that the two features with the highest absolute negative coefficient values are “ae\_” and “ae”, meaning that a discrepancy in the frequency of use of this feature is indeed an indicator that the authors are different or, to put it another way, that the frequency of use of these features is pretty stable in the production of an author, and it is thus a characteristic trait of an author (either because the author tends to use it a lot, or only rarely). This specific case might be connected with the transitional phase in medieval Latin where scholars started representing the diphthong “ae” with the single letter “e”, rather than with the two separate letters “ae”, as it was written throughout antiquity. A large difference in the frequency of use of these features might thus be an indication that the authors are different, one having a preference for “ae” and the other instead preferring “e”.

We can check that the ranked features are indeed useful for the classification by running an ablation experiment, also known as *Iterative Removal Of Features* (IROF) [243]. This approach consists of first assessing the performance of the classifier equipped with the entire feature set; then, sequentially, the feature with the highest absolute coefficient value is removed from the feature set (by setting the associated weights to zero in the classifier), and the performance of the classifier is reassessed (without re-training

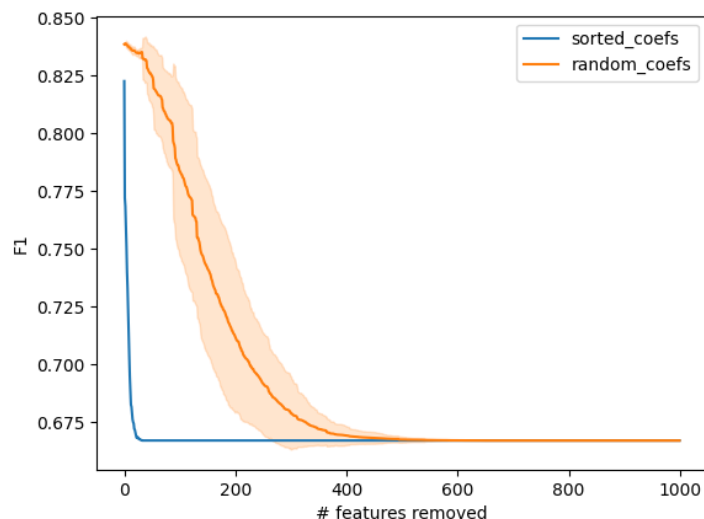


Figure 7.3: Results of the IROF validation on SVM classifier for SAV: we iteratively remove one feature at a time, following the descending order of the absolute values of the coefficients (`sorted_coefs`) or a random feature ranking (`random_coefs`). In particular, for the latter we show the mean  $F_1$  value obtained at the  $n$  feature removed for 10 random feature rankings, where the coloured shadow is the standard deviation.

the classifier).

The fact that the model performance drops as we iteratively remove features should come as no surprise. However, a good ranking of features that effectively reflects the features importance would cause the performance to degrade much faster (i.e., in less iterations) than any other uninformative ranking. This is shown in Figure 7.3, in which we compare the drop in performance as a function of the number of features removed, by considering our feature ranking (in blue) versus (10 trials of) a random ranking (in orange). The fact that the model becomes a dummy classifier after removing only a few features based on our ranking demonstrates that the importance criterion is indeed informative.

As already explained in Section 7.2.1, we can also employ the coefficients to obtain a form of local explanation, by multiplying the feature value extracted from a document by the correspondent coefficient. We randomly select 2 examples for `SameAuthor` and `DifferentAuthor`, and show the results in Figure 7.4. This visualisation highlights the biggest drawback of this XAI technique when applied to textual examples: if we limit the investigation to just a few features, we might risk to convey an incomplete, and thus wrong, picture, especially to a scholar that is not a ML expert. In fact, what we observe is that, on the basis of these examples, the outcome is often contradictory. First, many of the features displaying positive weights happen to be absent in the selected examples. Second, features displaying negative weights behave inconsistently across the examples, e.g., showing relatively high values (examples numbered 2 and 4), or very low values (examples numbered 1 and 3) regardless of their class labels (`SameAuthor` or `DifferentAuthor`). Hence, it seems clear that restricting the study to only a selected number of features is not enough to convey the full picture of the model behaviour.

Regarding this XAA approach, we can thus conclude that it contributes to justifying the decisions of the system in the eyes of the domain expert to some degree, but it is also rather problematic. As already noted, AAn tasks (as any other application of text classification) tend to be characterised by a high number of features, each one providing only a tiny contribution to the final classification decision. In other words, it is unlikely that there are just a handful of features that, by themselves, determine a classification decision. However, as we have shown, limiting the investigation to only a small portion of the features actually employed by the classifier incurs the risk to convey a picture that is just too narrow and simplistic. Thus, it is of primary importance to offer an analysis that includes the entirety of the

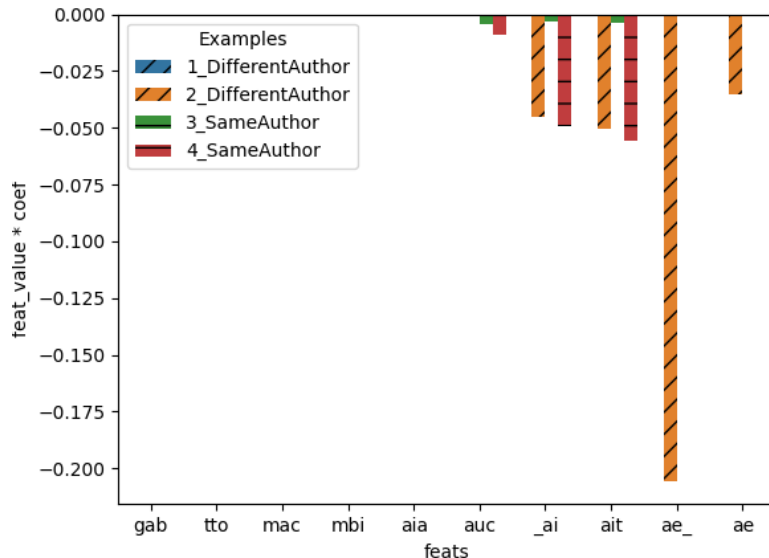


Figure 7.4: Local explanations for 4 examples in the test set, given the features listed in Table 7.2; we colour the SameAuthor class with the “—” pattern and the DifferentAuthor class with the “/” pattern. Note that the scores for many features is zero for all four examples.

feature set used by the classifier in the most user-friendly way possible, allowing a scholar to personalise and navigate the exploration to its full extent.

## 7.4.2 Probing the Transformer for AA

In our experiments, we train a simple LR model as a probe, since the classification head on top of the RoBERTa transformer has an equivalent complexity, and thus could not gain any more information from the latent representation. In fact, employing non-linear models as probes could be counterproductive: their accuracy might be caused by the memorisation of surface patterns, instead of the information actually captured by the latent representation [137]. We take the training set obtained in Section 7.3.1 and further split it in a stratified fashion into a training and test set for the probe, consisting of 90% and 10% of the instances respectively. The model hyperparameters are fine-tuned via 3-CV on the probe training set. The resulting model is then retrained on the full training set for the probe before evaluation. In our experiments, we probe the main model trained on the AA task. Note that we only show this method applied to the AA case, but it can be applied to AV and SAV tasks as well. However, probes for SAV should be handled carefully, since RoBERTa latent representation involves both texts.

As illustrated in Section 7.2.2, we test the transformer with five different probings:

- **POS  $n$ -grams**: we probe the model for POS  $n$ -grams, with  $n \in \{5, 10\}$ . In particular, the probe is asked to predict whether a certain POS  $n$ -grams is present ( $f(x_i) = 1$ ) or absent ( $f(x_i) = 0$ ) in the document; the labelling function  $f(x_i)$  is thus binary. We extract the POS tags via the `LatinCy` pipeline for the `SpaCy` library (see Appendix D).<sup>7</sup> We restrict the analysis to the 5 most discriminative POS  $n$ -grams in the corpus that best discriminate authors, for which the POS  $n$ -grams are evaluated via Chi-square (see Appendix A.2).
- **SQ  $n$ -grams**: the approach is equivalent to the POS  $n$ -grams probing, with  $n \in \{10, 15\}$ . We extract the syllabic quantities via the prosodic scanner in the `CLTK` library (see Appendix D).

<sup>7</sup>Documentation available at: <https://spacy.io/universe/project/latincy>.

- **Word lengths:** we probe whether the model takes the word-length distribution into account or not. In order to do so, we represent each document  $x_i$  by means of a histogram  $(b_i^{(1)}, b_i^{(2)}, \dots, b_i^{(B)})$ , in which the bin  $b_i^{(j)}$  accounts for the relative frequency of words of length  $j$  (i.e., the fraction of words of exactly  $j$  characters) in the document. Then, we cluster the documents thus represented in order to identify natural groups based on their word-length distribution; we use K-Means as our clustering algorithm and choose the optimal number of clusters via the Elbow method within the range  $[2, 10]$ . Each cluster is assigned a numerical ID, so that the labelling function  $f(x_i)$  is categorical in this case, and the probe is issued to label each document with the respective K-Means cluster ID. Note that the histogram representation is only used as a means for deciding the cluster to which each document belongs; that is, the probe is still trained and tested using the internal representations  $\phi(x_i)$  of the model.<sup>8</sup>
- **Function words:** we create a probe to check the extent to which the model learns from the frequency of use of the function words. To this aim, we apply a strategy that is similar to the aforementioned case for word lengths. That is, we first represent each document  $x_i$  as a histogram  $(b_i^{(w_1)}, b_i^{(w_2)}, \dots, b_i^{(w_B)})$ , in which the bin  $b_i^{(w_j)}$  accounts for the relative frequency of the function word  $w_j$  in  $x_i$ . We consider the list of 80 function words for Latin used in Corbara et al. [6], see Appendix C for the full list. As before, we label each document with the cluster ID to which it is assigned by the K-Means algorithm based on the histogram-based representations. The function  $f$  is thus again categorical.
- **Genre:** we probe the model for the genre of the documents; in particular, we ask the probe to classify the documents based on the sub-corpus they belong to, MEDLATINEPI or MEDLATINLIT. As such, we try to assess whether the transformer encodes the stylistic characteristics of documents of epistolary nature ( $f(x_i) = 1$ ) versus documents a different literary nature ( $f(x_i) = 0$ ); the labelling function  $f(x_i)$  is thus binary.

We show the results of the POS probing in the first portion of Table 7.3. The probes show high performance for all the POS  $n$ -grams considered, with the  $F_1$  performance always above 0.8, indicating that the transformer is likely learning from the syntax of the documents. These results are in line with the current literature on LM probing [189], and confirm that, even in AAn, models leverage POS  $n$ -grams in downstream tasks. On the other hand, the performance of the SQ probing, displayed in the second portion of Table 7.3, is much lower, with values between 0.6 and 0.7. However, these results indeed show knowledge of the concept of syllabic quantity by the transformer; this is an interesting discovery since, to our knowledge, this is the first work to explore this type of information in the latent space generated by a transformer.

Regarding word lengths and function words, we show the results of the two multi-class classifications in the first and second portion of Table 7.4 respectively; interestingly, the optimal number of clusters is 6 for both experiments. The probe show poor or mediocre results, getting higher scores in inferring

---

<sup>8</sup>A technical note: we use the implementation of K-Means provided by the `scikit-learn` library (see Appendix D), which relies on the Euclidean distance (aka L2) for computing the clusters. This turns out to be suboptimal in the case of word lengths, since the histograms actually represent ordered distributions, and since the L2 does not take into account the order of the dimensions of the feature vectors with which it operates. For example, the L2 distance between the pair of (normalised) vectors  $v_1 = (1, 0, 0, \dots, 0)$  and  $v_2 = (0, 1, 0, \dots, 0)$  is as large as the L2 distance between the same vector  $v_1$  and  $v_3 = (0, 0, 0, \dots, 1)$ , despite the fact that  $v_1$  and  $v_2$  represent documents that tend to use very short words while  $v_3$  instead represents a document that tends to use very long words. In order to counter this, in this case we represent our documents by means of cumulative distributions; in our example, this means that the distance between the (cumulative distributions)  $v'_1 = (1, 1, 1, \dots, 1)$  and  $v'_2 = (0, 1, 1, \dots, 1)$  turns out to be much smaller than the distance between  $v'_1$  and  $v'_3 = (0, 0, 0, \dots, 1)$ . A different solution would be to adopt, in place of the L2, a distance that is suited for ordinal data, such as the Wasserstein distance (aka Earth Mover Distance in computer science). However, we do not explore this possibility here, since the `scikit-learn` implementation does not allow to customise the distance function.

Table 7.3: POS and SQ probes results. Probes try to predict the presence of the given POS n-gram or SQ n-gram in the latent representation of the model. Note that for SQ we here employ the standard notation where “U” stands for a short syllable and “-” stands for a long syllable. We show the results in terms of accuracy (Acc), precision ( $P$ ), recall ( $R$ ) and  $F_1$ .

		<b>n-gram</b>	Acc	$P$	$R$	$F_1$
POS		adj noun adj noun verb	.825	.869	.825	.845
		adj noun noun adj noun	.882	.922	.882	.900
		adp noun adj noun verb	.821	.853	.821	.836
		noun adj noun adj noun	.873	.909	.873	.890
		noun adj noun verb verb	.853	.873	.853	.863
		<b>n-gram</b>	Acc	$P$	$R$	$F_1$
SQ		UUUUUU - UUUU	.670	.684	.670	.674
		UUUUUUUU - UU	.642	.647	.642	.644
		UUUUUUUUUU -	.654	.664	.654	.657
		UUUUUUUUUUUU	.601	.614	.601	.601
		UUUUUUUUUUUUU	.626	.670	.626	.639

Table 7.4: Word-lengths and function-words probes results. Probes try to predict the word-lengths cluster or function-words cluster in the latent representation of the model. The notational conventions are the same as in Table 7.3.

	#clusters	Acc	$P$	$R$	$F_1$
<b>Word lengths</b>	6	.487	.487	.487	.486
<b>Function words</b>	6	.617	.628	.617	.617

Table 7.5: Genre probes results. Probes try to predict the sub-corpus of the documents among MEDLATINEPI and MEDLATINLIT in the latent representation of the model. The notational conventions are the same as in Table 7.3.

	Acc	$P$	$R$	$F_1$
<b>Genre</b>	.979	.979	.979	.979

the function-words distribution of the documents. This highlights the importance of elements such as function words in the characterisation of literary authors [161]. We can hypothesise that the transformer’s apparent lack of encoding information regarding word-lengths distribution might stem from the authors sharing similar backgrounds, and thus similar vocabulary usages.

Regarding probing for the genre of the documents, the results are displayed in Table 7.5. The probe is clearly able to determine the sub-corpus the documents come from, thus indicating that the transformer indeed encodes the genre of the document into the latent space. This result could help warn the human expert against the risk that the neural model under investigation might be exploiting domain information, which should be avoided in AId studies (see Chapter 4). The classifier should focus on style-related information, and should not label a document as written by author  $A^*$  simply on the ground that  $A^*$  often writes in the same genre or topic as the document in question.

Summing up, this analysis could indeed reveal to a scholar some of the inner workings of a high-level model, by showing which features it leverages and which it avoids, thus reassuring the scholar of the outcome of the classification. In particular, the probing task would be well suited for an active interaction with the scholar who, prompted by their deep knowledge on the literary matter, could propose promising features to analyse, in the form of a ‘human-in-the-loop’ process. However, an important limitation, that scholars should be aware of, is that the probes can only be constructed around automatically decidable features, unless one wants to incur the cost of manually labelling the documents according to more complex features.

### 7.4.3 Factuals and counterfactuals for AV

In our experiments, we retrieve one factual and one counterfactual for both the SVM and RoBERTa models trained for the AV task. Note that we only exemplify this method as applied to the AV case; however, it can be applied to the SAV and AA tasks as well. In particular, we obtain the TfIdf vectors

(in the SVM case) or the encodings of the final hidden state (in the RoBERTa case) for a random test instance  $x$  and the entire training set; we then compute the Euclidean distances among  $x$  and all the training instances, and select the training instance closest to  $x$  which has the same (or different) label as the predicted label of  $x$ . Of course, both the number of (counter)factuals returned and the similarity measure to employ are parameters that can be modified.

The selected test example, which is an epistle from the author Pier della Vigna, is the following:

Fridericus uniuersis mundi *principibus* de *sinistris* rumoribus **Terrae Sanctae** Etsi *tam* iusta quam uehemens *causa* doloris et motus fuerit in nobis cum ad presentiam nostram **frater S.a uenerabili patre patriarcha** Antiocheno dilecto *amico nostro* presentium baiulus litterarum *accessit* ipsum *tamen* infeste uidere nequiuimus qui mittentem affectione quadam diligimus *singulari*. Uerum etiam tunc temporis cordis nostri *neruum* pertingerat rumor infestus et subitae nuntius tempestatis qui Coheminarum pestem ab originalibus sedibus Tartarea clade depulsam uelut molem *ingentem* per abrupta montium et decliuum fulminis ictibus *deuolutam* in **Sanctam Ciuitatem** irruisse crudeliter nuntiauit. Quae forte desolationis suae tempore habitatore continui solita defensari *cateruatim* undique concurrentibus *populis* colebatur dederatque cursui famosi *tamen* loci *longis* retro temporibus **Christicolis** maxime desiderata securitas et *sinistris* *auspiciis* diebus *illis* obtenta quarumdam occasione *treugarum* quas *soldanus* Damasci et Nathasar *soldanus* Craci qui *prius* *hostes* et *aduersarii* fuerant *concordiam* inuicem *facientes* ipsam cum **Christianis** ea condicione *fecerunt* quod tota regni Hierosolimitani terra quam **Christiani** possederant trans Iordanem retentissibi *uillis* et montanis aliquibus restituta **Christiani** *soldanis* eisdem in expugnatione *soldani* Babiloniae deberent assistere toto posse. Qua *confederatione* *tamquam* in sui perniciem inita *soldanus* accinctus *predictam* gentem *Barbaricam* Coheminarum per deserta uagantem et uelut feram in saltibus ante uenabulum fugientem ad suae defensionis *auxilium* conuocauit. *quibus* reputantes oblatum presidium potius quam petitum ad designata loca subito non minus *taciti* quam celeres perueniunt ut inuisos *hostes* aduenisse maturis nostrorum uigilantia nouerit quam uenturos. *sicque* factum est ut **Christianorum** exercitu cum soldanis predictis in guerram soldani Babiloniae apud Gazaram commorante **patriarcha** Hierosolimitanus de partibus Cismarinis ad partes *illas* athleta nouus *accessit*. [...]

It is the narration of an episode of the Crusades in the Holy Land, with the characteristics of historical chronicles. It is among the numerous letters written by the author during their station as chancellor of Emperor Frederick II.

The same factual example is selected for both the SVM and RoBERTa models; it is again an epistle, this time by author Giovanni Boccaccio:

Celeberrimi nominis *militi* Iacobo Pizinge serenissimi *principis* Federici Trinacrie regis logothete. *Generose miles* incertus mei Neapoli aliquamdiu fueram uere preterito. hinc enim plurimo desiderio trahebar redeundi in *patriam* quam autumpno nuper elapso indignans liqueram nec minus reuisendi libellos quos immeritos omiseram *sic* et *amicos* aliosque *caros*. inde uero urgebar ut consistere atque detinebar *nunca* uenerabili uiolentia nunc suasionibus nunc precibus incliti uiri Hugonis de *comitibus* **Sancti** Seuerini cuius credo splendidam famam noueris. Curabat enim uir eximius etiam me inuito totis uiribus ut me interueniente subsidio serenissime *domine* Iohanne Ierusalem et *sicilie regine* apud Parthenopeos *placido* locaret in otio. qua perplexitate *angebar* nimium nulla adhuc in parte satis firmato consilio. Et dum *sic* uariis agitarer curis quo pacto non memini factum *tamen* est ut ad aures deueniret meas uenerabile nomen **religiosi** hominis Ubertini de **ordine** Minorum **sacre theologie**

professoris et concuius tui cuius auditis meritis eumque ea tempestate Neapoli moram trahere pro quibusdam arduis tui suique *regis* in desiderium uenit **tam** conspicuum uidere uirum. **a** pueritia quippe mea etiam ultra tenelle etatis uires talium audivissimus fui. Nec mora. exhibiturus *reuerentiam* deb**itam** ad eum **accessi** atque adaperto **capite** primo paxillum miratus hominem quam **deuotissime** et humillime potui salutaui eum. Ipse autem graui quadam maturitate obuius factus me leta **facie** miti eloquio et morum laudabili comitate suscepit.

In this case, the selected factual is a letter to (ironically!) a notary of the Kingdom of Sicily; the epistle presents a first-person narration, describing some personal anecdotes happened during the author’s stay in Naples. The themes apparently could not be more different from the test example, but a closer inspection shows that the two texts share many references to religious orders and political relations. We demonstrate this by formatting in **bold** some of the former and in *italic* some of the latter.

Regarding the counterfactual, the same example is selected (again) for both SVM and RoBERTa; it is yet another epistle, this time by author Dante Alighieri (since in this experiment Dante plays the role of the positive class, while the other authors collectively play the role of the negative class):

Absita uiro predicante iustitiam ut per**essus** iniurias iniuriam inferentibus uelut benemerentibus pecuniam suam solu**at**. Non est hec uia rede**undi** ad *patriam* pater mi. sed si alia per uos ante**aut** deinde per alios inuenitur **que** fame Dantisque *honori* non derogat **illam** non lentis passibus acceptabo. quod si per nullam talem Florentia introitur nunquam Florentiam introibo. Quidni. nonne solis astrorumque specula ubique conspiciam. nonne dulcissimas ueritates potero speculari ubique sub celo ni **prius** *inglorium* ymo ignominiosum *populo* Florentino *ciuitati* me reddam. Quippe nec panis deficiet.

Unlike the factual, the counterfactual clearly has a very different domain than the test example, since it is a personal account of the tribulations the author experienced during his exile. Still, there are again some references to political concepts (again shown in *italic*).

All in all, it seems that both models are able to spot similarities and differences in the documents, especially the ones linked with the themes and references of the narration, highlighting similar textual patterns. Analogous to the results of the probing for the genre of the documents in Section 7.4.2, this could serve as a warning for the human experts examining the model, indicating the potential exploitation of domain-specific information, a practice to be avoided in AId research (see Chapter 4). However, the burden of spotting these similarities and differences is mainly on the human user, and this can be a difficult and time-consuming task. Coupling this XAI technique with other methods that highlight the textual regions, or features, that most determine the similarity among the documents, could be helpful in this sense. We give a very elementary exemplification of this by colouring in the texts the 10 *n*-grams that have the minimum differences among the feature values in the test example and in the factual (in **blue**), and among the feature values in the test example and in the counterfactual (in **red**) (the *n*-grams that are shared among all three texts are coloured in **violet**).

Other techniques that exist in the related literature generate ad-hoc synthetic examples as (counter) factuals (see for example Lampridis et al. [179]). While it might be possible in principle to generate synthetic textual instances for our case too, it is not clear how these examples could be useful for the human expert, who would realistically be interested in real-world textual documents only, and not in machine-generated texts.

## 7.5 Discussion

In this chapter, we have underlined the importance of explainability for authorship studies (eXplainable Authorship Analysis - XAAAn), with a specific focus on the case of cultural heritage, discussing the

**Problem 4** of this Thesis.

As we have shown, despite its importance, there are no existing XAI techniques specifically devised for authorship studies in the field of cultural heritage, nor for other applications of AAn. We thus experiment with three existing XAI methodologies proposed in other contexts (namely, feature ranking, probing, and factual and counterfactual selection), and we test them on the three main AId tasks, employing the medieval Latin dataset of Chapter 3 as case study. We make the code developed available to other researchers who might want to apply these techniques to other AAn problems.

In this study, we demonstrate that each XAI method tested contributes partially to clarifying the rationales behind the model’s predictions, and that they jointly provide some explanations of different aspects of the model. In particular, while feature ranking and probing shed light on the linguistic events leveraged by the model, (counter)factuals put these important linguistic events in context, showing real examples of the writing production under study.

However, we argue that the explanations that can be obtained with current, general-purpose techniques are still largely insufficient, even for a small-scale dataset with a limited number of authors, since they either convey a rather limited perspective of the inner working of the classifier (feature ranking) or heavily rely on the user’s input and intuition (probing, factuals and counterfactuals selection). Employing a combination of these methods, instead of using them in isolation, would mitigate, but not solve, this problem. In particular, a visualisation tool could help display the disputed document with the occurrences of the different features highlighted, with the highlighting coming in different shades depending on the class considered and the significance of the feature. Moreover, the tool could allow the domain expert to select a particular feature of interest, and show one or more examples from the dataset where the feature has a strong and significant presence. However, while solutions of this kind might help the expert navigate across relevant related cases, we argue they would still fall short of providing a convincing and conclusive explanation.

In future work, the exploration for methods that provide meaningful explanations for supporting the research of scholars should continue. We believe that aiming for concise and informative textual explanations (see for example Barratt [30], Le et al. [183]) is an avenue worth exploring, since this is the format most familiar to cultural heritage scholars.



## Chapter 8

# Conclusions

In this Ph.D. Thesis, we explored the four main issues in the AAn application to the cultural heritage domain that we have faced while investigating the AId problem of the *Epistle to Cangrande* (see Chapter 3). In each chapter, we endeavored to develop methodologies to address, if not resolve, one of these issues, conducting thorough investigations and extensive experimentation to robustly support our findings.

In Chapter 4, we evaluated the use of topic-agnostic features based on rhythmic elements to define authorial traits in Latin and Spanish (additionally, we have explored psycholinguistics-based features specifically for the latter). This approach aims to mitigate the influence of domain-related information on the AId analysis, since such information may lack reliability in terms of authorship. We showed that, while these feature sets might not be consistently advantageous in all the proposed settings, they nonetheless yield performance results comparable to state-of-the-art topic-aware methods, but with the significant advantage of being grounded in a more robust methodological basis. These results encourage expanding the analysis to other literary genres, such as theatrical pieces.

In Chapter 5, we set to explore ways to better exploit the authorial information contained in textual datasets of limited size, given the impossibility to otherwise increase the number of original documents at will. Thus, we analyzed the application of an alternative method (Diff-Vector, DV) for generating vectorial representations of textual documents, where a vector does not represent a single document, but a pair of documents. This representation appears to be indeed advantageous in most cases, since it either exploits more labelled instances from the same training data, or makes training more robust, than methods based on standard vectors. Indeed, the experiments showed that DVs bring substantial improvements, especially in low-resource tasks. In future work, it would be interesting to explore other functions to compute the DV feature values (aside from the absolute difference of the feature values of the two documents, employed in the present study), particularly by dynamically learning such functions from data.

In Chapter 6, we addressed the challenge of conducting effective AAn research in the presence of an adversarial agent attempting to deceive the authorial system. Our approach involved enhancing the robustness of the authorship classifier by incorporating synthetic samples generated by tailored automated systems into the training data. Despite the initially promising results, we consider this investigation unsuccessful: the observed improvements in classifier performance appear too inconsistent to warrant practical application. Nevertheless, we believe our contribution offers valuable insights into the limitations of current standard methodologies in generating useful samples for effective classification training, prompting the research into other directions. We underline the crucial importance of ensuring the generator adheres to the authorial style under consideration, and suggest that modifying pre-existing texts could be helpful in this regard.

Finally, in Chapter 7, we surveyed some standard XAI methods in order to assess their suitability

for authorship tasks in the cultural heritage setting, and their capacity to provide insightful explanations to a public of scholars. Our investigation revealed that each method examined contributes to partially elucidating the rationales behind the predictions of the authorship model. However, we also emphasized that the explanations provided by current general-purpose techniques remain largely inadequate: they either offer a limited perspective on the inner workings of the classifier, or depend heavily on the input and intuition of the user. Indeed, this analysis has been highly useful in highlighting the strengths and weaknesses of existing XAI methods, and in providing pointers for future research regarding more effective explanation techniques tailored to cultural heritage scholars researching AAn. In particular, we believe that crafting concise and informative textual explanations would be a promising research direction.

In this Thesis, we aimed to present a comprehensive examination of the applicability of AAn methods within the cultural heritage domain. We highlighted both the potential contributions these methods offer to the study of our history and the challenges researchers may encounter in such investigations. We delved into these problems, conducting a comprehensive analysis to deepen our understanding of these intricate phenomena, and proposed potential solutions to address said challenges. Although some projects and experiments in this work did not produce the anticipated positive results, we believe that the reported outcomes still provide meaningful contributions to the broader academic discourse. In conclusion, we hope that our research will offer valuable insights into the complexities of existing AAn methodologies and inspire future investigations in this field, opening new avenues for further exploration.

# Appendices

## Appendix A : Formulas

In this sections, we present the formulas of common algorithms that we employ in this Thesis.

### A.1 Evaluation metrics

In order to measure the performance of the classifiers, we employ various standard classification measures.

The “vanilla accuracy” (Acc) measure is defined as:

$$\text{Acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} \quad (1)$$

where TP, TN, FP, FN, represent the numbers of true positives, true negatives, false positives and false negatives generated by the classification system. Acc is thus the ratio between the number of correct predictions and the number of predictions.

The  $F_1$  measure is defined as:

$$F_1 = \begin{cases} \frac{2 \text{TP}}{2 \text{TP} + \text{FP} + \text{FN}} & \text{if } \text{TP} + \text{FP} + \text{FN} > 0 \\ 1 & \text{if } \text{TP} = \text{FP} = \text{FN} = 0 \end{cases} \quad (2)$$

$F_1$  ranges between 0 (worst) and 1 (best). For multi-class settings, we also employ its *macro-averaged* variant (denoted by  $F_1^M$ ) and its *micro-averaged* variant (denoted by  $F_1^\mu$ ):  $F_1^M$  is obtained by first computing the values of  $F_{1c}$  for each class  $c \in \mathcal{C}$ , and then averaging them;  $F_1^\mu$  is obtained by (i) computing the class-specific values  $\text{TP}_c, \text{FP}_c, \text{FN}_c$  for each class  $c \in \mathcal{C}$ , (ii) obtaining TP as the sum of all the  $\text{TP}_c$  (same for FP and FN), and then (iii) applying Equation 2. Note that, in single-label multi-class settings,  $F_1^\mu$  is equal to Acc, and it is also equal to micro-averaged precision and micro-averaged recall.

The  $K$  measure was first introduced by Sebastiani [260], and it is defined as:

$$K = \begin{cases} \frac{\text{TP}}{\text{TP} + \text{FN}} + \frac{\text{TN}}{\text{TN} + \text{FP}} - 1 & \text{if } (\text{TP} + \text{FN} > 0) \wedge (\text{TN} + \text{FP} > 0) \\ 2 \cdot \frac{\text{TN}}{\text{TN} + \text{FP}} - 1 & \text{if } (\text{TP} + \text{FN} = 0) \\ 2 \cdot \frac{\text{TP}}{\text{TP} + \text{FN}} - 1 & \text{if } (\text{TN} + \text{FP} = 0) \end{cases} \quad (3)$$

$K$  ranges from -1 (worst) to 1 (best), with 0 corresponding to the accuracy of the random classifier.

### A.2 Chi-square

The Chi-square test is used in statistics to test the independence of two events [299]. In our case, we employ it for feature selection, where the two events are the class label  $c \in \mathcal{C}$  and a certain feature  $t_k$ , and we seek to establish whether there is a correlation between the two; the stronger the correlation, the more probable would be for the feature to be selected. We use the Chi-square implementation provided

by the `scikit-learn` package (see Appendix D).<sup>1</sup> In particular, Chi-square is defined as:

$$\chi^2(t_k, c) = \frac{[\Pr(t_k, c) \Pr(\bar{t}_k, \bar{c}) - \Pr(t_k, \bar{c}) \Pr(\bar{t}_k, c)]^2}{\Pr(t_k) \Pr(\bar{t}_k) \Pr(c) \Pr(\bar{c})} \quad (4)$$

where probabilities are interpreted on the event space of documents; in other words,  $\Pr(t_k, c)$  represents the probability that, for a random document that belongs to class  $c$ , feature  $t_k$  appears in the document.

### A.3 TfIdf

Term frequency - Inverse document frequency (TfIdf) is a common weighting function for textual features in Information Retrieval, i.e., for attributing different degrees of importance to different features in the same document vector [249]. It is based on two major assumptions: first, the more a term appears in a document, the more said term is relevant for the document (the *term frequency* assumption); second, if a term is common in many documents in the collection, it is likely not particularly characteristic of the individual document (the *inverse document frequency* assumption). Weighting features via TfIdf (which is indeed an increasing function of relative frequency) is customary in AAn (see e.g., Ikae [138], Koppel et al. [174], Koppel and Winter [176], Menta and Garcia-Serrano [202]); in fact, the use of the inverse document frequency is justified by the fact that rare features (such as POS  $n$ -grams, word unigrams, or character  $n$ -grams) can be considered more indicative of style than common ones.

We employ TfIdf in order to assign a weight to character and word  $n$ -grams. We employ the smoothing “ltc” variant, as implemented in the `scikit-learn` package (see Appendix D),<sup>2</sup> where it is defined as:

$$\text{TfIdf}(t_k, d_i) = \text{Tf}(t_k, d_i) \cdot \text{Idf}(t_k, D) \quad (5)$$

where  $\text{TfIdf}(t_k, d_i)$  is the weight of feature  $t_k$  for document  $d_i$  in the dataset  $D$ , and

$$\text{Tf}(t_k, d_i) = \begin{cases} 1 + \log \text{Tf}(t_k, d_i) & \text{if } \text{Tf}(t_k, d_i) > 0 \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

where  $\text{Tf}(t_k, d_i)$  is the number of occurrences of feature  $t_k$  in document  $d_i$ , and

$$\text{Idf}(t_k, D) = \log \frac{1 + n}{1 + \text{df}(t_k)} + 1 \quad (7)$$

where  $n$  is the total number of documents in  $D$ , and  $\text{df}$  is the number of documents in  $D$  that contain the feature  $t_k$ .

The resulting TfIdf vectors are then normalized by the Euclidean norm.

## Appendix B : Datasets

In this section, we describe the various datasets we employ in this Thesis. Note that, depending on the specific project we employ them, different pre-processing steps might be taken, and they are explained in the related section.

<sup>1</sup>Documentation available at: [https://scikit-learn.org/stable/modules/generated/sklearn.feature\\_selection.chi2.html](https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.chi2.html).

<sup>2</sup>Documentation available at: [https://scikit-learn.org/stable/modules/generated/sklearn.feature\\_extraction.text.TfidfVectorizer.html](https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html), and [https://scikit-learn.org/stable/modules/generated/sklearn.feature\\_extraction.text.TfidfTransformer.html#sklearn.feature\\_extraction.text.TfidfTransformer](https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html#sklearn.feature_extraction.text.TfidfTransformer).

## B.1 MEDLATIN: MEDLATINEPI and MEDLATINLIT

This is the dataset we assembled in order to tackle the authorship problem of the *Epistle to Cangrande* (see Section 3). Since the *Epistle* is traditionally divided into two portions, Ep1 and Ep2, we built a separate dataset for each portion.

In both MEDLATINEPI and MEDLATINLIT Dante Alighieri is, of course, the author of some of the labelled texts. The texts attributed with certainty to the poet and written in Latin are few and well known; we have thus included all of them.<sup>3</sup> Concerning other authors, the approach we have chosen is to select literates who are as “close” (culturally and stylistically) to Dante Alighieri as possible, hence authors whose production is characterised by linguistic features similar to Alighieri’s [110]. The reason for this choice is that, if the non-Dantean texts used for training were very different from Dante’s training texts, any text even vaguely similar to Dante’s production would be recognised as Dantean, the classifier being untrained to make subtle distinctions. Instead, one can expect better results if the classifier is trained to spot minimal differences. We have thus done a large-scale screening of authors who have written in Latin around the same historical period of Dante’s, and who have written works of either epistolary or literary nature. Since the included authors are close to each other, in the above-mentioned cultural-stylistic sense, the two resulting datasets can be considered challenging ones for AAn systems.

The composition of the two datasets is described in detail in Tables 1 and 2.

MEDLATINEPI is composed of texts of epistolary genre (given that this is the nature of Ep1) mostly dating back to the 13th and 14th centuries, for a total of 294 epistles; the average length of these epistles is 378 words. Note that, concerning the epistles by Guido Faba and Pietro della Vigna (rows 4 and 5 of Table 1), we have not used the entire collections available from D’Angelo [319] and Gaudenzi [326], but only parts of them. One reason is that some of these epistles are extremely short in length (sometimes even a single sentence), and hence they would not have conveyed much information. The second reason is that, as it can be seen in Table 1, Guido Faba and Pietro della Vigna are the two authors for whom we have the highest number of epistles; including the collections in their entirety would have made the dataset even more imbalanced.

MEDLATINLIT contains instead texts of various genre, especially exegetic comments on literary works, chronicles and treatises (given the literary nature of Ep2), also mostly dating to the 13th and 14th centuries, for a total of 30 texts; the average length of these texts is 39,958 words, about 100 times longer (on average) than those of MEDLATINEPI. Some of these texts are not included in their entirety. In these cases, the portions excluded mainly consist of lengthy direct citations to other authors’ works.

The authorship of all of the texts included in the two datasets is certain, i.e., is not currently disputed by any scholar<sup>4</sup>. Some of the texts were already available in `.txt` format, and their inclusion in the dataset was thus fairly straightforward. Some other texts were only available in `.pdf` format, or only on paper; in these cases, we converted the `.pdf` or the scanned images into `.txt` format via an optical character recognition software,<sup>5</sup> and thoroughly corrected the output by hand.

We subjected all texts to a number of preprocessing steps necessary for performing accurate AAn. These steps are as follows:

- Removing any meta-textual information that has been inserted by the curator of the edition, such

---

<sup>3</sup>We have not included the *Quaestio de aqua et terra*, a work traditionally attributed to Dante Alighieri, exactly because its authorship is currently disputed. Other works by Alighieri, such as his masterpiece *Divina Commedia*, are not included because they are written not in Latin but in the Florentine vernacular, the language that would later form the basis of the Italian language.

<sup>4</sup>Note that from Petrus de Boateriis’ collection (see last row of Table 1) we have removed the epistle allegedly written by Cangrande della Scala to Henry VII, since it has recently been suggested that it may have been written by Dante Alighieri; this is the topic of Section 3.4.2.

<sup>5</sup>We have used `FreeOCR`, available at: <http://www.paperfile.net/>.

Table 1: Composition of the MEDLATINEPI dataset. The table shows the approximate historical period in which the texts were written (**Period**), the number of texts (**#d**) and the number of words (**#w**) that the collection consists of, respectively.

<b>Author</b>	<b>Text</b> (or collection)	<b>Period</b>	<b>#d</b>	<b>#w</b>	<b>Ed.</b>
Clara Assisiensis	<i>Epistola ad Ermentrudem</i>	1240-1253	1	249	[332]
	<i>Epistolae ad sanctam Agnetem de Praga I, II, III</i>	1234-1253	3	1,842	[332]
Dante Alighieri	Epistles	1304-1315	12	6,061	[322]
Giovanni Boccaccio	Epistles and letters	1340-1375	24	25,789	[310]
Guido Faba	Epistles	1239-1241	78	7,203	[326]
Pietro della Vigna	The collected epistles of Pietro della Vigna	1220-1249	146	65,004	[319]
(Various authors)	Epistles from the collection of Petrus de Boateriis	1250-1315	30	5,056	[339]

as titles, page numbers, quotation marks, square brackets, etc; this cleans the documents from obvious editorial intervention.

- Marking direct citations in Latin with asterisks, and direct citations in languages other than Latin (mostly Florentine vernacular) with curly brackets; this is both to allow ignoring them in the computation (since they are the production of someone different than the author of the text), or to use them as a potential authorial-related feature (for example, the usage of citations in different languages), at the discretion of the researcher.
- Replacing every occurrence of the character “v” with the character “u”.<sup>6</sup>

The two datasets are available for download at: <https://doi.org/10.5281/zenodo.4298503>; a **readme** file is also included, where we explain the structure of the archive.

<sup>6</sup>We do this in order to standardise the different approaches that editors might follow. In medieval written Latin, instead of the two modern graphemes “u-U” and “v-V”, there was only one grapheme, represented as a lowercase “u” and a capital “V”; some contemporary editors decide to follow this canon while others decide to modernise the written text with the two separate graphemes.

Table 2: Composition of the MEDLATINLIT dataset; we follow the same notation as in Table 1.

Author	Text	Period	#w	Ed.
Bene Florentinus	<i>Candelabrum</i>	1238	41,078	[309]
Benvenuto da Imola	<i>Comentum super Dantis Aldigherij Comediam</i>	1375-1380	105,096	[313]
	<i>Expositio super Valerio Maximo</i>	1380	3,419	[338]
	<i>Glose Bucolicorum Virgilii</i>	1380	3,912	[330]
Boncompagno da Signa	<i>Liber de obsidione Ancone</i>	1198-1200	7,821	[324]
	<i>Palma</i>	1198	5,022	[341]
	<i>Rota Veneris</i>	ante 1215	4,632	[323]
	<i>Ysagoge</i>	1204	8,550	[316]
Dante Alighieri	<i>De Vulgari Eloquentia</i>	1304-1306	11,384	[342]
	<i>Monarchia</i>	1313-1319	19,162	[334]
Filippo Villani	<i>Expositio seu comentum super Comedia Dantis Allegherii</i>	1391-1405	31,503	[321]
Giovanni Boccaccio	<i>De vita et moribus d. Francisci Petracchi</i>	1342	1,884	[320]
	<i>De mulieribus claris</i>	1361-1362	49,242	[344]
	<i>De Genealogia deorum gentilium</i>	1360-1375	198,508	[336]
Giovanni del Virgilio	<i>Allegorie super fabulas Ovidii Methamorphoseos</i>	1320	25,131	[317]
	<i>Ars dictaminis</i>	1320	2,376	[329]
Graziolo Bambaglioli	A Commentary on Dante's Inferno	1324	41,104	[337]
Guido da Pisa	<i>Expositiones et glose. Declaratio super Comediam Dantis</i>	1327-1328	87,822	[315]
Guido de Columnis	<i>Historia destructionis Troiae</i>	1272-1287	82,753	[328]
Guido Faba	<i>Dictamina rhetorica</i>	1226-1228	16,982	[327]
Iacobus de Varagine	<i>Chronica civitatis Ianuensis</i>	1295-1298	53,864	[333]
Iohannes de Appia	<i>Constitutiones Romandiolae</i>	1283	4,068	[312]
Iohannes de Plano Carpini	<i>Historia Mongalorum</i>	1247-1252	20,145	[318]
Iulianus de Spira	<i>Vita Sancti Francisci</i>	1232-1239	12,396	[332]
Nicola Trevet	<i>Expositio Herculis Furentis</i>	1315-1316	33,017	[343]
	<i>Expositio L. Annaei Senecae Agamemnonis</i>	1315-1316	19,873	[331]
Pietro Alighieri	<i>Comentum super poema Comedie Dantis</i>	1340-1364	186,608	[314]
Ryccardus de Sancto Germano	<i>Chronicon</i>	1216-1243	36,525	[325]
Raimundus Lullus	<i>Ars amativa boni</i>	1290	82,733	[335]
Zono de' Magnalis	Life of Virgilio	1340	2,136	[340]

## B.2 LATINITASANTIQUA

This dataset, composed of Latin prose texts, was originally assembled by us. To do so, we exploited the *Corpus Corporum* repository, developed by the University of Zurich,<sup>7</sup> and in particular its subsection

<sup>7</sup>Available at: <http://www.mlat.uzh.ch/MLS/>.

called *Latinitas Antiqua*, which contains various Latin works from the Perseus Digital library;<sup>8</sup> these works are meticulously tagged in XML. From this section, we further selected a group of texts that are (a) not poetry works, since our studies only deal with prose writings, and (b) not theatrical pieces, since they have a very peculiar format based on dialogue and scenes. The resulting dataset is presented in detail in Table 3. In total, it is composed of 90 prose texts by 25 Latin authors, spanning the Classical, Imperial and Early Medieval periods, and a variety of genres (mostly epistolary, historical, and rhetoric).

By exploiting the XML tagging, from each text we deleted foreign words (e.g., Greek) and the direct quotations from other authors, in order to retain only the “pure” production of the writer. We then removed every remaining XML tag.

Table 3: The *LatinitasAntiqua* dataset. The table shows, for each author, the number of entire texts available (#**d**), as divided in the online repository, their titles, and the collective sum of words (#**w**) contained after the pre-processing of the data.

Author	#d	Titles	#w
Ammianus Marcellinus	1	<i>Res gestae</i>	121,751
Apuleius	3	<i>Apologia, Florida, Metamorphoses</i>	82,401
Augustinus Hipponensis	1	<i>Epistularum Selectio</i>	46,225
Aulus Gellius	1	<i>Noctes Atticae</i>	79,710
Beda	1	<i>Historiam ecclesiasticam gentis Anglorum</i>	59,238
Cicero	29	<i>Academica, Brutus, Cato Maior: de Senectute, De Divinatione, De Fato, De Finibus, De Natura Deorum, De Officiis, De Optimo Genere Oratorum, De Oratore, De Republica, Epistolae ad Atticum, Epistolae ad Brutum, Epistolae ad Familiares, Epistolae ad Quintum, Laelius de Amicitia, Lucullus, Orationes (1, 2, 3, 4, 5, 6), Orator, Paradoxa stoicorum ad M. Brutum, Partitiones Oratoriae, Petitio consulatus, Topica, Tusculanae disputationes</i>	1,078,559
Columella	1	<i>De re rustica</i>	78,782
Cornelius Celsus	1	<i>De medicina</i>	102,459
Cornelius Nepos	1	<i>Vitae</i>	28,377
Cornelius Tacitus	5	<i>Annales, De Origine et Situ Germanorum Liber, De Vita Iulii Agricola, Dialogus de Oratoribus, Historiae</i>	161,937
Curtius Rufus	1	<i>Historiae Alexandri Magni</i>	74,260
Florus	1	<i>Epitome Rerum Romanorum</i>	26,237
Hieronymus Stridonensis	1	<i>Epistulae Selectiones</i>	43,692
Iulius Caesar	2	<i>De Bello Civili, De Bello Gallico</i>	83,628
Minucius Felix	1	<i>Octavius</i>	11,595
Plinius minor	1	<i>Epistolae</i>	64,881
Quintilianus	1	<i>Institutio Oratoria</i>	157,085

Continued on next page

<sup>8</sup>Available at: <http://www.perseus.tufts.edu/hopper/>.

Table 3 – continued from previous page

Author	#d	Titles	#w
Sallustius	2	<i>Bellum Iugurthinum, De Catilinae coniuratione</i>	31,887
Seneca	14	<i>Ad Lucilium Epistulae Morales, Apocolocyntosis, De Beneficiis, De Brevitate Vitae, De Clementia, De Consolatione ad Helvium, De Consolatione ad Marciam, De Consolatione ad Polybium, De Constantia, De Ira, De Otio, De Providentia, De Tranquillitate Animi, De Vita Beata</i>	234,328
Seneca maior	2	<i>Controversiae, Suasoriae</i>	75,058
Servius	3	<i>In Vergilii Aeneide comentarii, In Vergilii Eklogarum comentarii, In Vergilii Georgicis comentarii</i>	290,921
Sidonius Apollinaris	2	<i>Epistulae libri (1-7, 8-9)</i>	45,583
Suetonius	1	<i>De Vita Caesarum</i>	69,764
Titus Livius	13	<i>Ab Urbe Condita libri (1-2, 3-4, 5-7, 8-10, 21-22, 23-25, 26-27, 28-30, 31-34, 35-37, 38-39, 40-42, 43-45)</i>	486,263
Vitruvius	1	<i>De Architectura</i>	57,221

### B.3 KABALACORPUSA

This dataset was developed by Kabala [156]. In particular, of the four datasets that he assembled, we exploit CORPUSA, the biggest one, which consists of 39 texts by 22 authors from the 11th and 12th centuries. Long quotations and passages of poetry have been already removed from the texts by the author.

### B.4 PARLAMINT

The *Linguistically annotated multilingual comparable corpora of parliamentary debates ParlaMint.ana 2.1*<sup>9</sup> was developed and made publicly available by the digital infrastructure CLARIN; it contains the annotated transcriptions of many sessions of various European Parliaments. In particular, the Spanish repository covers the years 2015-2020.

### B.5 IMDB62

This dataset<sup>10</sup> was created and made publicly available (along with an extended version called IMDB1MILLION) by Seroussi et al. [262]. It contains movie reviews collected from the popular Internet Movie Database, and accounts for 62 authors/reviewers and 1,000 reviews authored by each of them.

<sup>9</sup>Available at: <https://www.clarin.si/repository/xmlui/handle/11356/1431>.

<sup>10</sup>Available at: [https://umlt.infotech.monash.edu/?page\\_id=266](https://umlt.infotech.monash.edu/?page_id=266).

## B.6 PAN11

This dataset<sup>11</sup> was created for the PAN-2011 international Authorship Identification competition [20]. The dataset is based on the Enron email corpus [172]. It contains three distinct problem settings (each coming with its own training, validation, and test set) for AV, where each training set contains emails by a single author, and each validation and test set contains a mixture of documents written by the training author and by others (not all are from the Enron corpus). Klimt and Yang [172] removed personal names and email addresses, and replaced them with specific tags; moreover, in order to reflect a natural task environment, some texts are not in English, or are automatically generated.

## B.7 VICTORIAN

This dataset<sup>12</sup> was created and made publicly available by Gungor [114]. It consists of books by American or British 18th-19th century novelists, subdivided into segments of 1,000 words each by the creator of the dataset, who also (i) removed the first and last 500 words of each book, and (ii) retained only the occurrences of the 10,000 words most frequent in the dataset, as a topic-filtering measure. The result is a corpus of more than 50,000 documents (i.e., segments) by 50 different authors. We restrict our attention to the “train” partition made available by the author (since the “test” partition does not contain the author labels), which accounts for 45 different authors. The corpus is imbalanced, with the least represented author accounting for 183 segments and the most represented one accounting for about 6,914 segments.

## B.8 ARXIV

This dataset, which we have created and made publicly available ourselves,<sup>13</sup> consists of abstracts of single-author papers from arXiv.<sup>14</sup> In order to limit domain-dependence, we harvested these abstracts by querying arXiv’s API with a list of computer-science-related keywords, mostly focused on machine learning.<sup>15</sup> Computer science articles are seldom written by a single author, which means that this dataset is not large. The corpus somehow follows a power-law distribution, with few prolific authors and many authors accounting for very few abstracts each: we retain authors with at least 10 abstracts to their name, resulting in a total of 1,469 documents from 100 authors. The 2 most prolific authors have 34 abstracts to their name, the 10 most prolific authors have written 22 or more, while 50% of the authors have no more than 12 abstracts to their name.

## B.9 TWEETFAKE

This dataset<sup>16</sup> was created and made publicly available by Fagni et al. [95]. It contains tweets from 17 human accounts and 23 bots, each one imitating one of the human accounts. The dataset is balanced (the tweets are half human- and half bot- generated) and already partitioned into a training, a validation, and a test set.

---

<sup>11</sup>Available at: <https://pan.webis.de/clef11/pan11-web/authorship-attribution.html>.

<sup>12</sup>Available at: <https://archive.ics.uci.edu/ml/datasets/Victorian+Era+Authorship+Attribution>.

<sup>13</sup>Available at: <https://doi.org/10.5281/zenodo.7404702>.

<sup>14</sup>Available at: <https://arxiv.org/>.

<sup>15</sup>The query used was: “deep learning, machine learning, information retrieval, computer science, data mining, support vector, logistic regression, artificial intelligence, supervised learning”.

<sup>16</sup>A limited version is available on Kaggle at: <https://www.kaggle.com/datasets/mtesconi/twitter-deep-fake-text>.

## B.10 EBG

The Extended Brennan-Greenstadt Corpus<sup>17</sup> was created by Brennan et al. [54]; we use the Obfuscation setting consisting of writings from 45 individuals contacted through the Amazon Mechanical Turk platform. Participants were asked to: i) upload examples of their own writing (of “scholarly” nature), and ii) to write a short essay regarding the description of their neighborhood while obscuring their writing style, without any specific instructions on how to do so.

## B.11 RJ

The Riddell-Juola Corpus<sup>18</sup> was created by Riddell et al. [242]; we use the Obfuscation setting. The documents were collected with the same policy as in the EBG corpus; however, in this corpus the participants were randomly assigned to receive the instruction to obfuscate their writing style.

## Appendix C : Latin function words

Through the Thesis, we employ the following list of 80 function words, if not otherwise stated: “a”, “ab”, “ac”, “ad”, “adhuc”, “ante”, “apud”, “atque”, “aut”, “autem”, “circa”, “contra”, “cum”, “de”, “dum”, “e”, “enim”, “ergo”, “et”, “etiam”, “ex”, “hec”, “iam”, “ibi”, “ideo”, “idest”, “igitur”, “in”, “inde”, “inter”, “ita”, “licet”, “nam”, “ne”, “nec”, “nisi”, “non”, “nunc”, “nunquam”, “ob”, “olim”, “per”, “post”, “postea”, “pro”, “propter”, “quando”, “quasi”, “que”, “quia”, “quidem”, “quomodo”, “quoniam”, “quoque”, “quot”, “satis”, “scilicet”, “sed”, “semper”, “seu”, “si”, “sic”, “sicut”, “sine”, “siue”, “statim”, “sub”, “super”, “supra”, “tam”, “tamen”, “tunc”, “ubi”, “uel”, “uelut”, “uero”, “uidelicet”, “unde”, “usque”, “ut”.

## Appendix D : Implementation details

The experiments presented here are implemented in the Python programming language. In particular, we extensively employ the following libraries:

- `scikit-learn` [224] (the documentation is available at: <https://scikit-learn.org/stable/>), for classic ML algorithms;
- `PyTorch` [222] (the documentation is available at: <https://pytorch.org/docs/stable/index.html>), for DL algorithms;
- `HuggingFace` (the documentation is available at: <https://huggingface.co/docs>), for pre-trained transformers models.

We also employ a series of libraries to pre-process the texts:

- `spaCy` [131], the documentation is available at: <https://spacy.io/usage>;
- `SciPy` [292], the documentation is available at: <https://docs.scipy.org/doc/scipy/>;
- `Natural Language Toolkit - NLTK` [44], the documentation is available at: <https://www.nltk.org/>;
- `Classical Language Toolkit - CLTK` [148], the documentation is available at: <http://cltk.org/>.

---

<sup>17</sup>Available on the Reproducible Authorship Attribution Benchmark Tasks (RAABT) on Zenodo: <https://zenodo.org/record/5213898#.YuuaNdJBzys>.

<sup>18</sup>Available on the same link of the EBG corpus at Footnote 17.



# Bibliography

## Papers published during this Ph.D. course

- [1] Corbara, S., Chulvi, B., Rosso, P., and Moreo, A. (2022a). Investigating topic-agnostic features for authorship tasks in Spanish political speeches. In *Natural Language Processing and Information Systems*, pages 394–402. Springer.
- [2] Corbara, S., Chulvi, B., Rosso, P., and Moreo, A. (2022b). Rhythmic and psycholinguistic features for authorship tasks in the Spanish Parliament: Evaluation and analysis. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction*, pages 79–92. Springer.
- [3] Corbara, S. and Moreo, A. (2023). Enhancing adversarial authorship verification with data augmentation. In *13th Italian Information Retrieval Workshop (IIR2023)*, pages 73–78.
- [4] Corbara, S. and Moreo, A. (2024). Forging the forger: An attempt to improve authorship verification via data augmentation. Submitted for publication to *IEEE Access*.
- [5] Corbara, S., Moreo, A., and Sebastiani, F. (2023a). Same or different? Diff-Vectors for authorship analysis. *ACM Transactions on Knowledge Discovery from Data*, 18(1):1–36.
- [6] Corbara, S., Moreo, A., and Sebastiani, F. (2023b). Syllabic quantity patterns as rhythmic features for Latin authorship attribution. *Journal of the Association for Information Science and Technology*, 74(1):128–141.
- [7] Corbara, S., Moreo, A., Sebastiani, F., and Tavoni, M. (2020). L’Epistola a Cangrande al vaglio della computazionale authorship verification: Risultati preliminari (con una postilla sulla cosiddetta “XIV Epistola di Dante Alighieri”). In *Atti del Seminario “Nuove Inchieste sull’Epistola a Cangrande”, 18 dicembre 2018, Pisa*, pages 153–192. Pisa University Press.
- [8] Corbara, S., Moreo, A., Sebastiani, F., and Tavoni, M. (2022c). MedLatinEpi and MedLatinLit: Two datasets for the computational authorship analysis of medieval Latin texts. *Journal on Computing and Cultural Heritage*, 15(3):1–15.
- [9] Setzu, M., Corbara, S., Monreale, A., Moreo, A., and Sebastiani, F. (2024). Explainable authorship identification in cultural heritage applications. *Journal on Computing and Cultural Heritage*, 17(3):1–23.

## Primary sources

- [10] Abbasi, A. and Chen, H. (2005). Applying authorship analysis to extremist-group web forum messages. *IEEE Intelligent Systems*, 20(5):67–75.
- [11] Adamovic, S., Miskovic, V., Milosavljevic, M., Sarac, M., and Veinovic, M. (2019). Automated language-independent authorship verification (for Indo-European languages). *Journal of the Association for Information Science and Technology*, 70(8):858–871.
- [12] Afroz, S., Brennan, M., and Greenstadt, R. (2012). Detecting hoaxes, frauds, and deception in writing style online. In *2012 IEEE Symposium on Security and Privacy*, pages 461–475. IEEE.
- [13] Afroz, S., Islam, A. C., Stolerman, A., Greenstadt, R., and McCoy, D. (2014). Doppelgänger finder: Taking stylometry to the underground. In *2014 IEEE Symposium on Security and Privacy*, pages 212–226. IEEE.
- [14] Aggarwal, C. C. (2014). Instance-based learning: A survey. In Aggarwal, C. C., editor, *Data Classification: Algorithms and Applications*, pages 157–185. CRC Press, London, UK.
- [15] Aggarwal, C. C. and Zhai, C., editors (2012). *Mining Text Data*. Springer, Heidelberg, DE.
- [16] Allred, J., Packer, S., Dozier, G., Aykent, S., Richardson, A., and King, M. C. (2020). Towards a human-AI hybrid for adversarial authorship. In *2020 SoutheastCon*, pages 1–8. IEEE.
- [17] Anwar, W., Bajwa, I. S., and Ramzan, S. (2019). Design and implementation of a machine learning-based authorship identification model. *Scientific Programming*, 2019.
- [18] Araujo-Pino, E., Gómez-Adorno, H., and Pineda, G. F. (2020). Siamese network applied to authorship verification. In *Working Notes of the 2020 Conference and Labs of the Evaluation Forum (CLEF 2020)*. CEUR-WS.org.
- [19] Argamon, S. (2008). Interpreting Burrows’s Delta: Geometric and probabilistic foundations. *Literary and Linguistic Computing*, 23(2):131–147.
- [20] Argamon, S. and Juola, P. (2011). Overview of the international authorship identification competition at PAN-2011. In Petras, V., Forner, P., and Clough, P. D., editors, *Notebook Papers of the 2011 Conference and Labs of the Evaluation Forum (CLEF 2011)*, volume 1177 of *CEUR Workshop Proceedings*. CEUR-WS.org.
- [21] Argamon, S., Koppel, M., Pennebaker, J. W., and Schler, J. (2009). Automatically profiling the author of an anonymous text. *Communications of the ACM*, 52(2):119–123.
- [22] Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223. PMLR.
- [23] Ascoli, A. R. (1997). Access to authority: Dante in the Epistle to Cangrande. In Baranski, Z. G., editor, *Seminario Dantesco Internazionale / International Dante Seminar 1*, pages 309–52. Le Lettere, Firenze.
- [24] Ashcroft, M., Johansson, F., Kaati, L., and Shrestha, A. (2016). Multi-domain alias matching using machine learning. In *2016 Third European Network Intelligence Conference (ENIC)*, pages 77–84. IEEE.

- [25] Ayer, M. (2014). *Allen and Greenough’s New Latin Grammar for Schools and College*. Dickinson College Commentaries, Carlisle, Pennsylvania.
- [26] Azarbyonad, H., Dehghani, M., Marx, M., and Kamps, J. (2015). Time-aware authorship attribution for short text streams. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 727–730.
- [27] Bagnall, D. (2015). Author identification using multi-headed recurrent neural networks. In Cappelato, L., Ferro, N., Jones, G. J. F., and SanJuan, E., editors, *Working Notes of the 2015 Conference and Labs of the Evaluation Forum (CLEF 2015)*, volume 1391 of *CEUR Workshop Proceedings*. CEUR-WS.org.
- [28] Baranski, Z. G. (1991). Comedia. Notes on Dante, the Epistle to Cangrande and medieval comedy. *Lectura Dantis*, (8):26–55.
- [29] Barlas, G. and Stamatatos, E. (2020). Cross-domain authorship attribution using pre-trained language models. In *Proceedings of the 16th Artificial Intelligence Applications and Innovations (AIAI 2020)*, pages 255–266. Springer.
- [30] Barratt, S. (2017). Interpnet: Neural introspection for interpretable deep learning. *arXiv preprint arXiv:1710.09511*.
- [31] Bartoli, A., Dagri, A., De Lorenzo, A., Medvet, E., and Tarlao, F. (2015). An author verification approach based on differential features. In *Working Notes of the 2015 Conference and Labs of the Evaluation Forum (CLEF 2015)*, Toulouse, FR.
- [32] Basile, C. and Lana, M. (2008). L’attribuzione di testi con metodi quantitativi: riconoscimento di testi gramsciani. *IDA informazioni : rivista di scienze dell’informazione*, 1/2:165–183.
- [33] Belinkov, Y. (2022). Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219.
- [34] Benedetto, D. and Degli Esposti, M. (2016). Dynamics of style and the case of the “Diario Postumo” by Eugenio Montale: A quantitative approach. In Degli Esposti, M., Altmann, E. G., and Pachet, F., editors, *Creativity and Universality in Language*, pages 157–176. Springer Nature, Cham, CH.
- [35] Berti, B., Esuli, A., and Sebastiani, F. (2023). Unravelling interlanguage facts via explainable machine learning. *Digital Scholarship in the Humanities*, 38(3):953–977.
- [36] Bevendorff, J., Chinea-Ríos, M., Franco-Salvador, M., Heini, A., Körner, E., Kredens, K., Mayerl, M., Pezik, P., Potthast, M., Rangel, F., et al. (2023). Overview of PAN 2023: Authorship verification, multi-author writing style analysis, profiling cryptocurrency influencers, and trigger detection. In *European Conference on Information Retrieval*, pages 518–526. Springer.
- [37] Bevendorff, J., Chulvi, B., Fersini, E., Heini, A., Kestemont, M., Kredens, K., Mayerl, M., Ortega-Bueno, R., Pezik, P., Potthast, M., et al. (2022). Overview of PAN 2022: Authorship verification, profiling irony and stereotype spreaders, and style change detection. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 13th International Conference of the CLEF Association, CLEF 2022, Bologna, Italy, September 5–8, 2022, Proceedings*, pages 382–394. Springer.
- [38] Bevendorff, J., Chulvi, B., la Peña Sarracén, G. L. D., Kestemont, M., Manjavacas, E., Markov, I., Mayerl, M., Potthast, M., Rangel, F., Rosso, P., Stamatatos, E., Stein, B., Wiegmann, M., Wolska, M., and Zangerle, E. (2021). Overview of PAN 2021: Authorship verification, profiling hate speech

- spreaders on twitter, and style change detection. In Candan, K. S., Ionescu, B., Goeuriot, L., Larsen, B., Müller, H., Joly, A., Maistro, M., Piroi, F., Faggioli, G., and Ferro, N., editors, *Experimental IR Meets Multilinguality, Multimodality, and Interaction - 12th International Conference of the CLEF Association, CLEF 2021, Virtual Event, September 21-24, 2021, Proceedings*, volume 12880 of *Lecture Notes in Computer Science*, pages 419–431. Springer.
- [39] Bevendorff, J., Ghanem, B., Giachanou, A., Kestemont, M., Manjavacas, E., Markov, I., Mayerl, M., Potthast, M., Pardo, F. M. R., Rosso, P., Specht, G., Stamatatos, E., Stein, B., Wiegmann, M., and Zangerle, E. (2020). Overview of PAN 2020: Authorship verification, celebrity profiling, profiling fake news spreaders on Twitter, and style change detection. In Arampatzis, A., Kanoulas, E., Tsikrika, T., Vrochidis, S., Joho, H., Lioma, C., Eickhoff, C., Névóel, A., Cappellato, L., and Ferro, N., editors, *Experimental IR Meets Multilinguality, Multimodality, and Interaction - 11th International Conference of the CLEF Association, CLEF 2020, Thessaloniki, Greece, September 22-25, 2020, Proceedings*, volume 12260 of *Lecture Notes in Computer Science*, pages 372–383. Springer.
- [40] Bevendorff, J., Hagen, M., Stein, B., and Potthast, M. (2019a). Bias analysis and mitigation in the evaluation of authorship verification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6301–6306.
- [41] Bevendorff, J., Potthast, M., Hagen, M., and Stein, B. (2019b). Heuristic authorship obfuscation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1098–1108.
- [42] Bevendorff, J., Stein, B., Hagen, M., and Potthast, M. (2019c). Generalizing unmasking for short texts. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 654–659.
- [43] Binongo, J. N. G. (2003). Who wrote the 15th book of Oz? An application of multivariate analysis to authorship attribution. *Chance*, 16(2):9–17.
- [44] Bird, S., Klein, E., and Loper, E. (2009). *Natural Language Processing with Python: Analyzing Text with the Natural Language ToolKit*. O’Reilly Media, Inc.
- [45] Bischoff, S., Deckers, N., Schliebs, M., Thies, B., Hagen, M., Stamatatos, E., Stein, B., and Potthast, M. (2020). The importance of suppressing domain style in authorship analysis. *arXiv preprint arXiv:2005.14714*.
- [46] Blas-Arroyo, J. L. (2003). “Perdóneme que se lo diga, pero vuelve usted a faltar a la verdad, señor Gonzalez”: Form and function of politic verbal behaviour in face-to-face Spanish political debates. *Discourse & Society*, 14(4):395–423.
- [47] Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022.
- [48] Boenninghoff, B., Hessler, S., Kolossa, D., and Nickel, R. M. (2019a). Explainable authorship verification in social media via attention-based similarity learning. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 36–45. IEEE.
- [49] Boenninghoff, B., Nickel, R. M., Zeiler, S., and Kolossa, D. (2019b). Similarity learning for authorship verification in social media. In *Proceedings of the 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2019)*, pages 2457–2461, Brighton, UK.

- [50] Bogdanova, A. and Romanov, V. (2021). Explainable source code authorship attribution algorithm. In *Journal of Physics: Conference Series*, volume 2134, page 012011. IOP Publishing.
- [51] Boran, T., Martinaj, M., and Hossain, M. S. (2020). Authorship identification on limited samplings. *Computers & Security*, 97:101943.
- [52] Boyd, R. L. (2018). Mental profile mapping: A psychological single-candidate authorship attribution method. *PloS one*, 13(7).
- [53] Boyd, R. L. and Pennebaker, J. W. (2015). Did Shakespeare write Double Falsehood? Identifying individuals by creating psychological signatures with text analysis. *Psychological science*, 26(5):570–582.
- [54] Brennan, M., Afroz, S., and Greenstadt, R. (2012). Adversarial stylometry: Circumventing authorship recognition to preserve privacy and anonymity. *ACM Transactions on Information and System Security (TISSEC)*, 15(3):1–22.
- [55] Brocardo, M. L., Traore, I., Saad, S., and Woungang, I. (2013). Authorship verification for short messages using stylometry. In *International Conference on Computer, Information and Telecommunication Systems (CITS)*, pages 1–6. IEEE.
- [56] Bromby, M. C. (2011). Juries and their understanding of forensic science: Are jurors equipped? *International Journal of Science in Society*, 2(2):247–256.
- [57] Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., and Shah, R. (1993). Signature verification using a “siamese” time delay neural network. *Advances in Neural Information Processing Systems*, 6.
- [58] Bull, P. and Wells, P. (2012). Adversarial discourse in Prime Minister’s questions. *Journal of Language and Social Psychology*, 31(1):30–48.
- [59] Burrows, J. (2002). “Delta”: A measure of stylistic difference and a guide to likely authorship. *Literary and Linguistic Computing*, 17(3):267–287.
- [60] Canettieri, P. (2016). Chi non ha scritto il Fiore. *Sulle Tracce Del Fiore*, pages 121–34.
- [61] Canettieri, P., Loreto, V., Rovetta, M., and Santini, G. (2008). Philology and information theory. *Cognitive philology*, 1.
- [62] Carvalho, D. V., Pereira, E. M., and Cardoso, J. S. (2019). Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8:832.
- [63] Casadei, A. (2016). Sempre contro l’autenticità dell’Epistola a Cangrande. *Studi danteschi*, LXXXI:215–46.
- [64] Casadei, A., editor (2020). *Atti del Seminario “Nuove Inchieste sull’Epistola a Cangrande”*. Pisa University Press, Pisa, IT.
- [65] Castro, D. C., Arcia, Y. A., Brioso, M. P., and Muñoz, R. (2015). Authorship verification, average similarity analysis. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 84–90.
- [66] Cañete, J., Chaperon, G., Fuentes, R., Ho, J.-H., Kang, H., and Pérez, J. (2020). Spanish pre-trained BERT model and evaluation data. In *PML4DC at ICLR 2020*.
- [67] Ceccarelli, L. (2018). *Prosodia e metrica latina classica, con cenni di metrica greca*. Società Editrice Dante Alighieri, Roma, IT.

- [68] Cerra, D., Datcu, M., and Reinartz, P. (2014). Authorship analysis based on data compression. *Pattern Recognition Letters*, 42:79–84.
- [69] Chandrasekaran, R. and Manimannan, G. (2012). Use of probabilistic neural network in the classification of articles of ambiguous authorship. *International Journal of Engineering Research & Technology (IJERT)*, 1(7).
- [70] Chaski, C. E. (2005). Who’s at the keyboard? Authorship attribution in digital evidence investigations. *International Journal of Digital Evidence*, 4(1):1–13.
- [71] Chhatwal, R., Gronvall, P., Huber-Fliflet, N., Keeling, R., Zhang, J., and Zhao, H. (2018). Explainable text classification in legal document review: A case study of explainable predictive coding. In *2018 IEEE International Conference on Big Data*, pages 1905–1911. IEEE.
- [72] Cho, K., van Merriënboer, B., Bahdanau, D., and Bengio, Y. (2014). On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111.
- [73] Chulvi, B., Molpeceres, M., Rodrigo, M. F., Toselli, A. H., and Rosso, P. (2023). Politicization of immigration and language use in political elites: A study of Spanish Parliamentary speeches. *Journal of Language and Social Psychology*, 43(2).
- [74] Cooper, F. A., Antoniak, M., De Sa, C., Migiel, M., and Mimno, D. (2021). “Tecnologica cosa”: Modeling storyteller personalities in Boccaccio’s “Decameron”. In *Proceedings of the 5th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 147–153.
- [75] Corbara, S. (2019). The Epistle to Cangrande through the lens of computational authorship verification. Master’s thesis, University of Pisa, Pisa, IT.
- [76] Corbara, S., Moreo, A., Sebastiani, F., and Tavoni, M. (2019). The Epistle to Cangrande through the lens of computational authorship verification. In *Proceedings of the 1st International Workshop on Pattern Recognition for Cultural Heritage (PatReCH 2019)*, pages 148–158, Trento, IT.
- [77] Coulthard, M. (2013). On admissible linguistic evidence. *Journal of Law and Policy*, 21(2).
- [78] Crammer, K. and Singer, Y. (2001). On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2(Dec):265–292.
- [79] Custódio, J. E. and Paraboni, I. (2018). EACH-USP ensemble cross-domain authorship attribution. *Working Notes of the 2018 Conference and Labs of the Evaluation Forum (CLEF 2018)*.
- [80] Dahllöf, M. (2012). Automatic prediction of gender, political affiliation, and age in Swedish politicians from the wording of their speeches – A comparative study of classifiability. *Literary and linguistic computing*, 27(2):139–153.
- [81] Danilevsky, M., Qian, K., Aharonov, R., Katsis, Y., Kawas, B., and Sen, P. (2020). A survey of the state of explainable AI for natural language processing. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing (AAACL/IJCNLP 2020)*, pages 447–459, Suzhou, CN.
- [82] Dauber, E., Caliskan, A., Harang, R., and Greenstadt, R. (2018). Git blame who? Stylistic authorship attribution of small, incomplete source code fragments. In *Proceedings of the 40th International Conference on Software Engineering: Companion Proceedings*, pages 356–357.

- [83] De Vel, O., Anderson, A., Corney, M., and Mohay, G. (2001). Multi-topic e-mail authorship attribution forensics. In *Proceedings ACM Conference on Computer Security-Workshop on Data Mining for Security Applications*.
- [84] Ding, S. H. H., Fung, B. C. M., Iqbal, F., and Cheung, W. K. (2017). Learning stylometric representations for authorship analysis. *IEEE Transactions on Cybernetics*, 49(1):107–121.
- [85] Domingos, P. M. and Pazzani, M. J. (1996). Beyond independence: Conditions for the optimality of the simple Bayesian classifier. In *Proceedings of the 13th International Conference on Machine Learning (ICML 1996)*, pages 105–112, Bari, IT.
- [86] Donahue, D. and Rumshisky, A. (2018). Adversarial text generation without reinforcement learning. *arXiv preprint arXiv:1810.06640*.
- [87] Dumalus, A. and Fernandez, P. (2011). Authorship attribution using writer’s rhythm based on lexical stress. In *Proceedings of the 11th Philippine computing science congress (PCSC 2011)*. Computing Society of the Philippines.
- [88] Dumpala, S. H., Chakraborty, R., and Kopparapu, S. K. (2018). A novel data representation for effective learning in class imbalanced scenarios. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI 2018)*, pages 2100–2106, Stockholm, SE.
- [89] Eddington, D. (2004). *Spanish Phonology and Morphology*. Studies in Functional and Structural Linguistics. John Benjamins Publishing Company, Amsterdam, NL.
- [90] Eder, M. (2011). Style-markers in authorship attribution: A cross-language study of the authorial fingerprint. *Studies in Polish Linguistics*, 6(1).
- [91] Esuli, A., Moreo, A., and Sebastiani, F. (2019). Funnelling: A new ensemble method for heterogeneous transfer learning and its application to cross-lingual text classification. *ACM Transactions on Information Systems*, 37(3).
- [92] Evert, S., Proisl, T., Jannidis, F., Reger, I., Pielström, S., Schöch, C., and Vitt, T. (2017). Understanding and explaining Delta measures for authorship attribution. *Digital Scholarship in the Humanities*, 32(suppl\_2):ii4–ii16.
- [93] Ezen-Can, A. (2020). A comparison of LSTM and BERT for small corpus. *arXiv preprint arXiv:2009.05451*.
- [94] Fabien, M., Villatoro-Tello, E., Motlicek, P., and Parida, S. (2020). BertAA: BERT fine-tuning for authorship attribution. In *Proceedings of the 17th International Conference on Natural Language Processing (ICON)*, pages 127–137.
- [95] Fagni, T., Falchi, F., Gambini, M., Martella, A., and Tesconi, M. (2021). TweepFake: About detecting deepfake tweets. *Plos one*, 16(5).
- [96] Faust, C., Dozier, G., Xu, J., and King, M. C. (2017). Adversarial authorship, interactive evolutionary hill-climbing, and AuthorCAAT-III. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8. IEEE.
- [97] Fenton-Smith, B. (2008). Discourse structure and political performance in adversarial parliamentary wuestioning. *Journal of Language and Politics*, 7(1):97–118.

- [98] Fernández-Cabana, M., Rúas-Araújo, J., and Alves-Pérez, M. T. (2014). Psicología, lenguaje y comunicación: Análisis con la herramienta LIWC de los discursos y tweets de los candidatos a las elecciones gallegas de 2012. *Anuario de Psicología*, 44(2):169–184.
- [99] Flach, P. A. (2017). Classifier calibration. In Sammut, C. and Webb, G. I., editors, *Encyclopedia of Machine Learning*, pages 212–219. Springer, Heidelberg, DE, 2nd edition.
- [100] Forstall, C. and Scheirer, W. (2010). Features from frequency: Authorship and stylistic analysis using repetitive sound. *Journal of the Chicago Colloquium on Digital Humanities and Computer Science*, 1(2):1–23.
- [101] Forstall, C. W., Jacobson, S. L., and Scheirer, W. J. (2011). Evidence of intertextuality: Investigating Paul the Deacon’s *Angustae Vitae*. *Literary and Linguistic Computing*, 26(3):285–296.
- [102] Frantzeskou, G., Stamatatos, E., Gritzalis, S., Chaski, C. E., and Howald, B. S. (2007). Identifying authorship by byte-level n-grams: The Source Code Author Profile (SCAP) method. *International Journal of Digital Evidence*, 6(1):1–18.
- [103] Gamon, M. (2004). Linguistic correlates of style: Authorship classification with deep linguistic analysis features. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, pages 611–617. International Committee on Computational Linguistics.
- [104] García-Díaz, J. A., Colomo-Palacios, R., and Valencia-García, R. (2022). Psychographic traits identification based on political ideology: An author analysis study on Spanish politicians’ tweets posted in 2020. *Future Generation Computer Systems*, 130:59–74.
- [105] Gaston, J., Narayanan, M., Dozier, G., Cothran, D. L., Arms-Chavez, C., Rossi, M., King, M. C., and Xu, J. (2018). Authorship attribution vs. Adversarial authorship from a LIWC and sentiment analysis perspective. In *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 920–927. IEEE.
- [106] Ginzburg, C. (1989). *Clues: Roots of an Evidential Paradigm*, pages 96–125. Johns Hopkins University Press.
- [107] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press, Cambridge, US.
- [108] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. *Advances in Neural Information Processing Systems*, 27.
- [109] Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.
- [110] Grieve, J. (2007). Quantitative authorship attribution: An evaluation of techniques. *Literary and Linguistic Computing*, 22(3):251–270.
- [111] Gu, Y., Mishra, B. D., and Clark, P. (2021). DREAM: Uncovering mental models behind language models. *arXiv preprint arXiv:2112.08656*.
- [112] Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., and Pedreschi, D. (2018). A survey of methods for explaining black box models. *ACM Computing Surveys*, 51(5):1–42.
- [113] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. (2017). Improved training of Wasserstein GANs. *Advances in Neural Information Processing Systems*, 30.

- [114] Gungor, A. (2018). Benchmarking authorship attribution techniques using over a thousand books by fifty Victorian era novelists. Master’s thesis, Department of Computer and Information Science, Purdue University, Indianapolis, US.
- [115] Guo, Y., Greiner, R., and Schuurmans, D. (2005). Learning coordination classifiers. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI 05)*, pages 714–721, Edinburgh, UK.
- [116] Gutierrez, J., Casillas, J., Ledesma, P., Fuentes, G., and Meza, I. (2015). Homotopy based classification for author verification task. In Cappellato, L., Ferro, N., Jones, G. J. F., and SanJuan, E., editors, *Working Notes of the 2015 Conference and Labs of the Evaluation Forum (CLEF 2015)*, volume 1391 of *CEUR Workshop Proceedings*. CEUR-WS.org.
- [117] Hall, R. G. and Sowell, M. U. (1989). “Cursus” in the Can Grande epistle: A forger shows his hand? *Lectura Dantis*, (5):89–104.
- [118] Halvani, O. (2021). *Practice-oriented Authorship Verification*. PhD thesis, Technische Universität Darmstadt, Darmstadt, DE.
- [119] Halvani, O., Graner, L., and Regev, R. (2020). TAVeer: An interpretable topic-agnostic authorship verification method. In Volkamer, M. and Wressnegger, C., editors, *Proceedings of the 15th International Conference on Availability, Reliability and Security (ARES)*, pages 1–10. ACM.
- [120] Halvani, O., Graner, L., and Vogel, I. (2018). Authorship verification in the absence of explicit features and thresholds. In *Advances in Information Retrieval: 40th European Conference on IR Research, ECIR 2018, Grenoble, France, March 26-29, 2018, Proceedings 40*, pages 454–465. Springer.
- [121] Halvani, O., Winter, C., and Graner, L. (2019). Assessing the applicability of authorship verification methods. In *Proceedings of the 14th International Conference on Availability, Reliability and Security (ARES)*, pages 1–10. ACM.
- [122] Halvani, O., Winter, C., and Pflug, A. (2016). Authorship verification for different languages, genres and topics. *Digital Investigation*, 16:S33–S43.
- [123] Hamon, R., Junklewitz, H., and Sanchez, I. (2020). Robustness and explainability of artificial intelligence. Technical Report EUR 30040, Publications Office of the European Union, Luxembourg, LU.
- [124] Harris, S. (2001). Being politically impolite: Extending politeness theory to adversarial political discourse. *Discourse & society*, 12(4):451–472.
- [125] Harrison, R. (2021). Cogitatorium. <http://rharriso.sites.truman.edu/>. Accessed: 07-07-2021.
- [126] Hatua, A., Mukherjee, A. M., and Verma, R. (2021). On the feasibility of using GANs for claim verification-experiments and analysis. In *Proceedings of the 2021 Workshop on Reducing Online Misinformation Through Credible Information Retrieval*.
- [127] Hirst, G. and Wei Feng, V. (2012). Changes in style in authors with Alzheimer’s disease. *English Studies*, 93(3):357–370.
- [128] Holmes, D. I. (1998). The evolution of stylometry in humanities scholarship. *Literary and Linguistic Computing*, 13(3):111–117.
- [129] Holmes, D. I. and Forsyth, R. S. (1995). The Federalist revisited: New directions in authorship attribution. *Literary and Linguistic computing*, 10(2):111–127.

- [130] Holmes, D. I., Robertson, M., and Paez, R. (2001). Stephen Crane and the New-York Tribune: A case study in traditional and non-traditional authorship attribution. *Computers and the Humanities*, 35:315–331.
- [131] Honnibal, M., Montani, I., Van Landeghem, S., and Boyd, A. (2020). spaCy: Industrial-strength natural language processing in Python.
- [132] Hooker, G. and Mentch, L. (2019). Please stop permuting features: An explanation and alternatives. *arXiv preprint arXiv:1905.03151*.
- [133] Hoover, D. L. and Hess, S. (2009). An exercise in non-ideal authorship attribution: The mysterious Maria Ward. *Literary and Linguistic Computing*, 24(4):467–489.
- [134] Hosseinia, M. and Mukherjee, A. (2018). Experiments with neural networks for small and large scale authorship verification. *arXiv preprint arXiv:1803.06456*.
- [135] Hu, Z., Lee, R. K.-W., Aggarwal, C. C., and Zhang, A. (2022). Text Style Transfer: A review and experimental evaluation. *ACM SIGKDD Explorations Newsletter*, 24(1):14–45.
- [136] Huertas-Tato, J., Martín, A., and Camacho, D. (2023). Using authorship embeddings to understand writing style in social media. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 60–71. Springer.
- [137] Hwitt, J. and Liang, P. (2019). Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 2733–2743, Hong Kong, CN.
- [138] Ikae, C. (2021). UniNE at PAN-CLEF 2021: Authorship verification. In *Working Notes of the 2021 Conference and Labs of the Evaluation Forum (CLEF 2021)*, pages 1995–2003, Bucharest, RO.
- [139] Iqbal, F., Binsalleeh, H., Fung, B. C. M., and Debbabi, M. (2013). A unified data mining solution for authorship analysis in anonymous textual communications. *Information Sciences*, 231:98–112.
- [140] Ivanov, L., Aebig, A., and Meerman, S. (2018). Lexical stress-based authorship attribution with accurate pronunciation patterns selection. In *International Conference on Text, Speech, and Dialogue*, pages 67–75. Springer.
- [141] Jafariakinabad, F., Tarnpradab, S., and Hua, K. A. (2020). Syntactic neural model for authorship attribution. In Barták, R. and Bell, E., editors, *The Thirty-Third International FLAIRS Conference*, pages 234–239. AAAI Press.
- [142] Jankowska, M., Milios, E., and Keselj, V. (2014). Author verification using common n-gram profiles of text documents. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 387–397.
- [143] Janson, T. (1975). *Prose Rhythm in Medieval Latin From the 9th to the 13th Century*. Almqvist & Wiksell International, Stockholm, SE.
- [144] Jenaro-MacLennan, L. (1974). *The Trecento commentaries on the Divina Commedia and the Epistle to Cangrande*. Clarendon Press, Oxford.
- [145] Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning (ECML 1998)*, pages 137–142, Chemnitz, DE.

- [146] Jockers, M. L. and Witten, D. M. (2010). A comparative study of machine learning methods for authorship attribution. *Literary and Linguistic Computing*, 25(2):215–223.
- [147] Johansson, F., Kaati, L., and Shrestha, A. (2013). Detecting multiple aliases in social media. In *2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013)*, pages 1004–1011. IEEE.
- [148] Johnson, K. P., Burns, P., Stewart, J., and Cook, T. (2021). CLTK: The Classical Language Toolkit. <https://github.com/cltk/cltk>.
- [149] Jones, K., Nurse, J. R. C., and Li, S. (2022). Are you Robert or RoBERTa? Deceiving online authorship attribution models using neural text generators. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 16, pages 429–440.
- [150] Jordan, K. N., Sterling, J., Pennebaker, J. W., and Boyd, R. L. (2019). Examining long-term trends in politics and culture through language of political leaders and cultural institutions. *Proceedings of the National Academy of Sciences*, 116(9):3476–3481.
- [151] Jullien, M., Valentino, M., and Freitas, A. (2022). Do transformers encode a foundational ontology? Probing abstract classes in natural language. *arXiv preprint arXiv:2201.10262*.
- [152] Juola, P. (2006). Authorship attribution. *Foundations and Trends in Information Retrieval*, 1(3):233–334.
- [153] Juola, P. (2012). Stylometry and immigration: A case study. *JL & Pol’y*, 21:287.
- [154] Juola, P. (2013). How a computer program helped reveal JK Rowling as author of *a cuckoo’s calling*. *Scientific American*, 20:13.
- [155] Juola, P. (2019). Artificial intelligence and pattern evidence: A legal application for AI. In *XX Simposio Argentino de Inteligencia Artificial (ASAI 2019)-JAIIO 48 (Salta)*.
- [156] Kabala, J. (2020). Computational authorship attribution in medieval Latin corpora: The case of the Monk of Lido (ca. 1101–08) and Gallus Anonymous (ca. 1113–17). *Language Resources and Evaluation*, 54(1):25–56.
- [157] Karras, T., Laine, S., and Aila, T. (2019). A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410.
- [158] Keeline, T. and Kirby, T. (2019). “Auceps Syllabarum”: A digital analysis of Latin prose rhythm. *The Journal of Roman Studies*, 109:161–204.
- [159] Kenton, J. D. M.-W. C. and Toutanova, L. K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.
- [160] Kešelj, V., Peng, F., Cercone, N., and Thomas, C. (2003). N-gram-based author profiles for authorship attribution. In *Proceedings of the Conference Pacific Association for Computational Linguistics (PACLING)*, volume 3, pages 255–264.
- [161] Kestemont, M. (2014). Function words in authorship attribution. From black magic to theory? In *Proceedings of the 3rd Workshop on Computational Linguistics for Literature (CLFL)*, pages 59–66. ACL.

- [162] Kestemont, M., Manjavacas, E., Markov, I., Bevendorff, J., Wiegmann, M., Stamatatos, E., Potthast, M., and Stein, B. (2020). Overview of the cross-domain authorship verification task at PAN 2020. In Cappellato, L., Eickhoff, C., Ferro, N., and Névéal, A., editors, *Working Notes of the 2020 Conference and Labs of the Evaluation Forum (CLEF 2020)*, volume 2696 of *CEUR Workshop Proceedings*. CEUR-WS.org.
- [163] Kestemont, M., Manjavacas, E., Markov, I., Bevendorff, J., Wiegmann, M., Stamatatos, E., Stein, B., and Potthast, M. (2021). Overview of the cross-domain authorship verification task at PAN 2021. In *Working Notes of the 2021 Conference and Labs of the Evaluation Forum (CLEF 2021)*, volume 2936 of *CEUR Workshop Proceedings*. CEUR-WS.org.
- [164] Kestemont, M., Martens, G., and Ries, T. (2019a). A computational approach to authorship verification of Johann Wolfgang Goethe’s contributions to the *Frankfurter gelehrte Anzeigen* (1772–73). *Journal of European Periodical Studies*, 4(1).
- [165] Kestemont, M., Moens, S., and Deploige, J. (2015). Collaborative authorship in the twelfth century: A stylometric study of Hildegard of Bingen and Guibert of Gembloux. *Digital Scholarship in the Humanities*, 30(2):199–224.
- [166] Kestemont, M., Stamatatos, E., Manjavacas, E., Daelemans, W., Potthast, M., and Stein, B. (2019b). Overview of the cross-domain authorship attribution task at PAN 2019. In Cappellato, L., Ferro, N., Losada, D. E., and Müller, H., editors, *Working Notes of the 2019 Conference and Labs of the Evaluation Forum (CLEF 2019)*, volume 2380 of *CEUR Workshop Proceedings*. CEUR-WS.org.
- [167] Kestemont, M., Stover, J., Koppel, M., Karsdorp, F., and Daelemans, W. (2016). Authenticating the writings of Julius Caesar. *Expert Systems with Applications*, 63:86–96.
- [168] Kestemont, M., Tschuggnall, M., Stamatatos, E., Daelemans, W., Specht, G., Stein, B., and Potthast, M. (2018). Overview of the author identification task at PAN-2018: Cross-domain authorship attribution and style change detection. In Cappellato, L., Ferro, N., Nie, J.-Y., and Soulier, L., editors, *Working Notes of the 2018 Conference and Labs of the Evaluation Forum (CLEF 2018)*, volume 2125 of *CEUR Workshop Proceedings*. CEUR-WS.org.
- [169] Khonji, M. and Iraqi, Y. (2014). A slightly-modified GI-based author-verifier with lots of features (ASGALF). *Working Notes of the 2014 Conference and Labs of the Evaluation Forum (CLEF 2014)*, 1180:977–983.
- [170] Kilgour, R. I., Gray, A. R., Sallis, P. J., and Macdonell, S. G. (1998). A fuzzy logic approach to computer software source code authorship analysis. In *Proceedings of the Fourth International Conference on Neural Information Processing*. Springer-Verlag.
- [171] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- [172] Klimt, B. and Yang, Y. (2004). The Enron Corpus: A new dataset for email classification research. In *Proceedings of the 15th European Conference on Machine Learning (ECML 2004)*, pages 217–226, Pisa, IT.
- [173] Kobayashi, S. (2018). Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 2, pages 452–457. ACL.

- [174] Koppel, M., Schler, J., and Argamon, S. (2009). Computational methods in authorship attribution. *Journal of the American Society for information Science and Technology*, 60(1):9–26.
- [175] Koppel, M., Schler, J., and Bonchek-Dokow, E. (2007). Measuring differentiability: Unmasking pseudonymous authors. *Journal of Machine Learning Research*, 8(6):1261–1276.
- [176] Koppel, M. and Winter, Y. (2014). Determining if two documents are written by the same author. *Journal of the Association for Information Science and Technology*, 65(1):178–187.
- [177] Kusner, M. J. and Hernández-Lobato, J. M. (2016). GANs for sequences of discrete elements with the Gumbel-softmax distribution. *arXiv preprint arXiv:1611.04051*.
- [178] Lagutina, K., Lagutina, N., Boychuk, E., Larionov, V., and Paramonov, I. (2021). Authorship verification of literary texts with rhythm features. In *28th Conference of Open Innovations Association (FRUCT)*, pages 240–251. IEEE.
- [179] Lampridis, O., State, L., Guidotti, R., and Ruggieri, S. (2022). Explaining short text classification with diverse synthetic exemplars and counter-exemplars. *Machine Learning*, pages 1–34.
- [180] Larner, S. (2014). *Forensic Authorship Analysis and the World Wide Web*. Springer.
- [181] Layton, R., Watters, P., and Ureche, O. (2013). Identifying faked hotel reviews using authorship analysis. In *2013 Fourth Cybercrime and Trustworthy Computing Workshop*, pages 1–6. IEEE.
- [182] Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196. PMLR.
- [183] Le, T., Wang, S., and Lee, D. (2020). GRACE: Generating concise and informative contrastive sample to explain neural network model’s prediction. In *Proceedings of the 26th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD 2020)*, pages 238–248.
- [184] Leonard, R. A., Ford, J. E. R., and Christensen, T. K. (2017). Forensic linguistics: Applying the science of linguistics to issues of the law. *Hofstra Law Review*, 45(3):11.
- [185] Lertvittayakumjorn, P. and Toni, F. (2019). Human-grounded evaluations of explanation methods for text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 5195–5205, Hong Kong, CN.
- [186] Lin, B. Y., Lee, S., Khanna, R., and Ren, X. (2020). Birds have four legs?! NumerSense: Probing numerical commonsense knowledge of pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*, pages 6862–6868.
- [187] Linardatos, P., Papastefanopoulos, V., and Kotsiantis, S. (2020). Explainable AI: A review of machine learning interpretability methods. *Entropy*, 23(1):18.
- [188] Liu, H., Yin, Q., and Wang, W. Y. (2019a). Towards explainable NLP: A generative explanation framework for text classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 5570–5581, Firenze, IT.
- [189] Liu, N. F., Gardner, M., Belinkov, Y., Peters, M. E., and Smith, N. A. (2019b). Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2019)*, pages 1073–1094, Minneapolis, US.

- [190] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019c). RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.
- [191] Loshchilov, I. and Hutter, F. (2018). Decoupled weight decay regularization. In *Proceedings of the 6th International Conference on Learning Representations (ICLR 2018)*, Vancouver, CA.
- [192] Love, H. (2002). *Attributing Authorship: An Introduction*. Cambridge University Press.
- [193] Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., and Lee, S.-I. (2020). From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, 2(1):56–67.
- [194] Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NIPS 2017)*, pages 4765–4774, Long Beach, US.
- [195] Lutosławski, W. (1897). La loi stylométrique. *Comptes rendus des séances de l'Académie des Inscriptions et Belles-Lettres*, 41(3):311–314.
- [196] Luyckx, K. and Daelemans, W. (2008). Authorship attribution and verification with many authors and limited data. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 513–520. COLING.
- [197] Manjavacas, E., De Gussem, J., Daelemans, W., and Kestemont, M. (2017). Assessing the stylistic properties of neurally generated text in authorship attribution. In *Proceedings of the Workshop on Stylistic Variation*, pages 116–125.
- [198] Manousakis, N. and Stamatatos, E. (2018). Devising “Rhesus”: A strange “collaboration” between Aeschylus and Euripides. *Digital Scholarship in the Humanities*, 33(2):347–361.
- [199] McCarthy, R. and O’Sullivan, J. (2021). Who wrote Wuthering Heights? *Digital Scholarship in the Humanities*, 36(2):383–391.
- [200] McNemar, Q. (1947). Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.
- [201] Mendenhall, T. C. (1887). The characteristic curves of composition. *Science*, 9(214):237–249.
- [202] Menta, A. and Garcia-Serrano, A. (2021). Authorship verification with neural networks via stylistic feature concatenation. In *Working Notes of the 2021 Conference and Labs of the Evaluation Forum (CLEF 2021)*. CEUR-WS.org.
- [203] Michailidis, P. D. (2022). A scientometric study of the stylometric research field. In *Informatics*, volume 9, page 60. MDPI.
- [204] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
- [205] Moreo, A., Esuli, A., and Sebastiani, F. (2016). Lightweight random indexing for polylingual text classification. *Journal of Artificial Intelligence Research*, 57:151–185.
- [206] Moreo, A., Esuli, A., and Sebastiani, F. (2018). Revisiting distributional correspondence indexing: A Python reimplementation and new experiments. *arXiv preprint arXiv:1810.09311*.

- [207] Moreo, A., Esuli, A., and Sebastiani, F. (2020). Learning to weight for text classification. *IEEE Transactions on Knowledge and Data Engineering*, 32(2):302–316.
- [208] Mosteller, F. and Wallace, D. L. (1963). Inference in an authorship problem: A comparative study of discrimination methods applied to the authorship of the disputed Federalist Papers. *Journal of the American Statistical Association*, 58(302):275–309.
- [209] Muttenthaler, L., Lucas, G., and Amann, J. (2019). Authorship attribution in fan-fictional texts given variable length character and word n-grams. In Cappellato, L., Ferro, N., Losada, D. E., and Müller, H., editors, *Working Notes of the 2019 Conference and Labs of the Evaluation Forum (CLEF 2019)*, volume 2380 of *CEUR Workshop Proceedings*. CEUR-WS.org.
- [210] Nardi, B. (1990). Il punto sull’Epistola a Cangrande. In “*Lecturae*” e altri studi danteschi, pages 205–25. Le Lettere, Firenze.
- [211] Navarro, B. (2015). A computational linguistic approach to Spanish Golden Age sonnets: Metrical and semantic aspects. In *Proceedings of the Fourth Workshop on Computational Linguistics for Literature*, pages 105–113.
- [212] Neidorf, L., Krieger, M. S., Yakubek, M., Chaudhuri, P., and Dexter, J. P. (2019). Large-scale quantitative profiling of the Old English verse tradition. *Nature human behaviour*, 3(6):560–567.
- [213] Nguyen, D., Gravel, R., Trieschnigg, D., and Meder, T. (2013). “How old do you think I am?” A study of language and age in Twitter. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 7, pages 439–448.
- [214] Niculescu-Mizil, A. and Caruana, R. (2005a). Obtaining calibrated probabilities from boosting. In *Proceedings of the 21st Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI 2005)*, pages 413–420, Arlington, US.
- [215] Niculescu-Mizil, A. and Caruana, R. (2005b). Predicting good probabilities with supervised learning. In *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005)*, pages 625–632, Bonn, DE.
- [216] Nini, A. (2018). An authorship analysis of the Jack the Ripper letters. *Digital Scholarship in the Humanities*, 33(3):621–636.
- [217] Nisbett, R. E., Peng, K., Choi, I., and Norenzayan, A. (2001). Culture and systems of thought: Holistic versus analytic cognition. *Psychological review*, 108(2):291.
- [218] Oberhelman, S. M. and Hall, R. G. (1984). A new statistical analysis of accentual prose rhythms in imperial Latin authors. *Classical Philology*, 79(2):114–130.
- [219] Ochab, J. K. and Essler, H. (2019). Stylometry of literary papyri. In *Proceedings of the 3rd International Conference on Digital Access to Textual Cultural Heritage*, page 139–142, New York, NY, USA. Association for Computing Machinery.
- [220] Ortega-Bueno, R., Chulvi, B., Rangel, F., Rosso, P., and Fersini, E. (2022). Profiling irony and stereotype spreaders on twitter (irostereo). In *Working Notes of the 2022 Conference and Labs of the Evaluation Forum (CLEF 2022)*, volume 3180.
- [221] Overdorf, R. and Greenstadt, R. (2016). Blogs, Twitter feeds, and Reddit comments: Cross-domain authorship attribution. *Proceedings on Privacy Enhancing Technologies*, 2016(3):155–171.

- [222] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.
- [223] Pavlopoulos, J. and Konstantinidou, M. (2023). Computational authorship analysis of the Homeric poems. *International Journal of Digital Humanities*, 5(1):45–64.
- [224] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.
- [225] Pellegrini, P. (2018). La quattordicesima epistola di Dante Alighieri: Primi appunti per una attribuzione. *Studi di Erudizione e di Filologia Italiana*, 7:5–20.
- [226] Pennebaker, J. W., Boyd, R. L., Jordan, K., and Blackburn, K. (2015). The development and psychometric properties of LIWC2015. Technical report, The University of Texas at Austin.
- [227] Pennebaker, J. W., Chung, C. K., Frazee, J., Lavergne, G. M., and Beaver, D. I. (2014). When small words foretell academic success: The case of college admissions essays. *PLoS one*, 9(12).
- [228] Perkins, R. (2015). Native language identification (NLID) for forensic authorship analysis of weblogs. In Dawson, M. and Omar, M., editors, *New Threats and Countermeasures in Digital Crime and Cyber Terrorism*, pages 213–234. IGI Global, Hershey, US.
- [229] Platt, J. C. (2000). Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In Smola, A., Bartlett, P., Schölkopf, B., and Schuurmans, D., editors, *Advances in Large Margin Classifiers*, pages 61–74. The MIT Press, Cambridge, MA.
- [230] Plecháč, P. (2021). Relative contributions of Shakespeare and Fletcher in Henry VIII: An analysis based on most frequent words and most frequent rhythmic patterns. *Digital Scholarship in the Humanities*, 36(2):430–438.
- [231] Polignano, M., de Gemmis, M., and Semeraro, G. (2020). Contextualized BERT sentence embeddings for author profiling: The cost of performances. In *Proceedings of the 2020 Computational Science and Its Applications (ICCSA 2020)*, pages 135–149. Springer.
- [232] Potha, N. and Stamatatos, E. (2014). A profile-based method for authorship verification. In *Artificial Intelligence: Methods and Applications: 8th Hellenic Conference on AI, SETN 2014, Ioannina, Greece, May 15-17, 2014. Proceedings 8*, pages 313–326. Springer.
- [233] Potha, N. and Stamatatos, E. (2017). An improved Impostors Method for authorship verification. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 8th International Conference of the CLEF Association, CLEF 2017, Dublin, Ireland, September 11–14, 2017*, pages 138–144. Springer.
- [234] Potha, N. and Stamatatos, E. (2018). Intrinsic author verification using topic modeling. In *Proceedings of the 10th Hellenic Conference on Artificial Intelligence*, pages 1–7.
- [235] Potha, N. and Stamatatos, E. (2019). Dynamic ensemble selection for author verification. In *European Conference on Information Retrieval*, pages 102–115. Springer.

- [236] Potthast, M., Hagen, M., and Stein, B. (2016). Author obfuscation: Attacking the state of the art in authorship verification. *Working Notes of the 2016 Conference and Labs of the Evaluation Forum (CLEF 2016)*, pages 716–749.
- [237] Potthast, M., Kiesel, J., Reinartz, K., Bevendorff, J., and Stein, B. (2018). A stylometric inquiry into hyperpartisan and fake news. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 231–240.
- [238] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.
- [239] Rajagopal, D., Balachandran, V., Hovy, E. H., and Tsvetkov, Y. (2021). SELFEXPLAIN: A self-explaining architecture for neural text classifiers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP 2021)*, pages 836–850, Punta Cana, DO.
- [240] Rangel, F., Rosso, P., Montes-y Gómez, M., Potthast, M., and Stein, B. (2018). Overview of the 6th author profiling task at PAN 2018: Multimodal gender identification in twitter. In *Working Notes of the 2018 Conference and Labs of the Evaluation Forum (CLEF 2018)*, pages 1–38.
- [241] Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). “Why should I trust you?” Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2016)*, pages 1135–1144, San Francisco, US.
- [242] Riddell, A., Wang, H., and Juola, P. (2021). A call for clarity in contemporary authorship attribution evaluation. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 1174–1179.
- [243] Rieger, L. and Hansen, L. K. (2020). IROF: A low-resource evaluation metric for explanation methods. In *Proceedings of the ICLR 2020 Workshop on AI for Affordable Healthcare*, Addis Ababa, ET.
- [244] Rocha, A., Scheirer, W. J., Forstall, C. W., Cavalcante, T., Theophilo, A., Shen, B., Carvalho, A., and Stamatatos, E. (2017). Authorship attribution for social media forensics. *IEEE Transactions on Information Forensics and Security*, 12(1):5–33.
- [245] Ruder, S., Ghaffari, P., and Breslin, J. G. (2016). Character-level and multi-channel Convolutional Neural Networks for large-scale authorship attribution. *arXiv preprint arXiv:1609.06686*.
- [246] Rudman, J. (1998). Non-traditional authorship attribution studies in the *Historia Augusta*: Some caveats. *Literary and Linguistic Computing*, 13(3):151–157.
- [247] Saedi, C. and Dras, M. (2021). Siamese networks for large-scale author identification. *Computer Speech & Language*, 70:101241.
- [248] Salminen, J., Kandpal, C., Kamel, A. M., Jung, S.-g., and Jansen, B. J. (2022). Creating and detecting fake reviews of online products. *Journal of Retailing and Consumer Services*, 64.
- [249] Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523.
- [250] Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2019). DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. In *NeurIPS EMC<sup>2</sup> Workshop*.

- [251] Sapkota, U., Bethard, S., Montes, M., and Solorio, T. (2015). Not all character n-grams are created equal: A study in authorship attribution. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 93–102.
- [252] Sapkota, U., Solorio, T., Montes, M., Bethard, S., and Rosso, P. (2014). Cross-topic authorship attribution: Will out-of-topic data help? In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1228–1237.
- [253] Sari, Y., Stevenson, M., and Vlachos, A. (2018). Topic or style? Exploring the most useful features for authorship attribution. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING 2018)*, pages 343–353, Santa Fe, US.
- [254] Sasso, G. (2013). Sull’Epistola a Cangrande. *La Cultura*, (3):359–446.
- [255] Savoy, J. (2013). The Federalist Papers revisited: A collaborative attribution scheme. *Proceedings of the American Society for Information Science and Technology*, 50(1):1–8.
- [256] Savoy, J. (2019). Authorship of Pauline epistles revisited. *Journal of the Association for Information Science and Technology*, 70(10):1089–1097.
- [257] Schmid, M. R., Iqbal, F., and Fung, B. C. M. (2015). E-mail authorship attribution using customized associative classification. *Digital Investigation*, 14:S116–S126.
- [258] Scolari, F. (1819). *Note ad alcuni luoghi delli primi cinque canti della Divina Commedia*. Tip. Picotti, Venezia.
- [259] Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys (CSUR)*, 34(1):1–47.
- [260] Sebastiani, F. (2015). An axiomatically derived measure for the evaluation of classification algorithms. In *Proceedings of the 2015 International Conference on the Theory of Information Retrieval*, pages 11–20.
- [261] Seidman, S. (2013). Authorship verification using the Impostors method. In *Working Notes of the 2013 Conference and Labs of the Evaluation Forum (CLEF 2013)*, pages 23–26.
- [262] Seroussi, Y., Zukerman, I., and Bohnert, F. (2014). Authorship attribution with topic models. *Computational Linguistics*, 40(2):269–310.
- [263] Shrestha, P., Sierra, S., González, F. A., Montes, M., Rosso, P., and Solorio, T. (2017). Convolutional neural networks for authorship attribution of short texts. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, volume 2, pages 669–674.
- [264] Sidorov, G. O. (2018). Automatic authorship attribution using syllables as classification features. *Rhema*, (1):62–81.
- [265] Solan, L. M. (2012). Intuition versus algorithm: The case of forensic authorship attribution. *JL & Pol’y*, 21:551.
- [266] Sommerschild, T., Assael, Y., Pavlopoulos, J., Stefanak, V., Senior, A., Dyer, C., Bodel, J., Prag, J., Androutopoulos, I., and de Freitas, N. (2023). Machine learning for ancient languages: A survey. *Computational Linguistics*, pages 1–44.

- [267] Spinazzè, L. (2014). “Cursus in Clausula”: An online analysis tool of Latin prose. In Tomasi, F., Turco, R. R. D., Tammaro, A. M., and Buzzetti, D., editors, *Proceedings of the Third AIUCD Annual Conference on Humanities and Their Methods in the Digital Ecosystem, AIUCD 2014, Bologna, Italy, September 18-19, 2014*, pages 10:1–10:6. ACM.
- [268] Stamatatos, E. (2009). A survey of modern authorship attribution methods. *Journal of the American Society for Information Science and Technology*, 60(3):538–556.
- [269] Stamatatos, E. (2016). Authorship verification: A review of recent advances. *Research in Computing Science*, 123:9–25.
- [270] Stamatatos, E. (2018). Masking topic-related information to enhance authorship attribution. *Journal of the Association for Information Science and Technology (JASIST)*, 69(3):461–473.
- [271] Stamatatos, E., Daelemans, W., Verhoeven, B., Juola, P., López-López, A., Potthast, M., and Stein, B. (2015). Overview of the author identification task at pan 2015. In Cappellato, L., Ferro, N., Jones, G. J. F., and SanJuan, E., editors, *Working Notes of the 2015 Conference and Labs of the Evaluation Forum (CLEF 2015)*, volume 1391 of *CEUR Workshop Proceedings*. CEUR-WS.org.
- [272] Stamatatos, E., Daelemans, W., Verhoeven, B., Potthast, M., Stein, B., Juola, P., Sanchez-Perez, M. A., and Barrón-Cedeño, A. (2014). Overview of the author identification task at PAN 2014. In *Working Notes of the 2014 Conference and Labs of the Evaluation Forum (CLEF 2014)*, pages 1–21.
- [273] Stamatatos, E., Kestemont, M., Kredens, K., Pezik, P., Heini, A., Bevendorff, J., Stein, B., and Potthast, M. (2022). Overview of the authorship verification task at PAN 2022. In *CEUR workshop proceedings*, volume 3180, pages 2301–2313.
- [274] Stein, B., Lipka, N., and Prettenhofer, P. (2011). Intrinsic plagiarism analysis. *Language Resources and Evaluation*, 45:63–82.
- [275] Stein, B., Lipka, N., and zu Eissen, S. M. (2008). Meta analysis within authorship verification. In *19th International Workshop on Database and Expert Systems Applications*, pages 34–39. IEEE.
- [276] Stover, J. A., Winter, Y., Koppel, M., and Kestemont, M. (2016). Computational authorship verification method attributes a new work to a major 2nd century African author. *Journal of the Association for Information Science and Technology*, 67(1):239–242.
- [277] Sturtevant, E. H. (1922). Syllabification and syllabic quantity in Greek and Latin. *Transactions and Proceedings of the American Philological Association*, 53:35–51.
- [278] Tang, X., Liang, S., and Liu, Z. (2019). Authorship attribution of the “golden lotus” based on text classification methods. In *Proceedings of the 2019 3rd International Conference on Innovation in Artificial Intelligence*, pages 69–72.
- [279] Tearle, M., Taylor, K., and Demuth, H. (2008). An algorithm for automated authorship attribution using neural networks. *Literary and linguistic computing*, 23(4):425–442.
- [280] Theophilo, A., Giot, R., and Rocha, A. (2023). Authorship attribution of social media messages. *IEEE Transactions on Computational Social Systems*, 10(1):10–23.
- [281] Theophilo, A., Padilha, R., Andaló, F. A., and Rocha, A. (2022). Explainable artificial intelligence for authorship attribution on social media. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2022)*, pages 2909–2913, Singapore, SN.

- [282] Tognetti, G. (1982). *Criteri per la trascrizione di testi medievali latini e italiani*, volume 51 of *Quaderni della Rassegna degli Archivi di Stato*. Ministero per i beni culturali e ambientali, Roma, IT.
- [283] Toynbee, P. (1918). Dante and the Cursus: A new argument in favour of the authenticity of the “Quaestio de Aqua et Terra”. *The Modern Language Review*, 13(4):420–430.
- [284] Tuccinardi, E. (2017). An application of a profile-based method for authorship verification: Investigating the authenticity of Pliny the Younger’s letter to Trajan concerning the Christians. *Digital Scholarship in the Humanities*, 32(2):435–447.
- [285] Tukey, J. W. (1949). Comparing individual means in the analysis of variance. *Biometrics*, pages 99–114.
- [286] Tuzzi, A. and Cortelazzo, M. A. (2018). *Drawing Elena Ferrante’s Profile: Workshop Proceedings, Padova, 7 September 2017*. Padova UP.
- [287] Uchendu, A., Le, T., Shu, K., and Lee, D. (2020). Authorship attribution for neural text generation. In *Conference on Empirical Methods in Natural Language Processing 2020 (EMNLP)*, pages 8384–8395. ACL.
- [288] Vainio, R., Välimäki, R., Hella, A., Kaartinen, M., Immonen, T., Vesanto, A., and Ginter, F. (2019). Reconsidering authorship in the Ciceronian corpus through computational authorship attribution. *Ciceroniana On Line*, 3(1).
- [289] van der Goot, R., Ljubešić, N., Matroos, I., Nissim, M., and Plank, B. (2018). Bleaching text: Abstract features for cross-lingual gender prediction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 383–389. ACL.
- [290] Van Halteren, H., Baayen, H., Tweedie, F., Haverkort, M., and Neijt, A. (2005). New machine learning methods demonstrate the existence of a human stylome. *Journal of Quantitative Linguistics*, 12(1):65–77.
- [291] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.
- [292] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, 17:261–272.
- [293] Weerasinghe, J., Singh, R., and Greenstadt, R. (2021). Feature vector difference based authorship verification for open world settings. In *Working Notes of the 2021 Conference and Labs of the Evaluation Forum (CLEF 2021)*, Bucharest, RO.
- [294] Wiegreffe, S. and Pinter, Y. (2019). Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 11–20, Hong Kong, CN.

- [295] Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5(2):241–259.
- [296] Wu, H., Zhang, Z., and Wu, Q. (2021). Exploring syntactic and semantic features for authorship attribution. *Applied Soft Computing*, 111.
- [297] Wu, T.-F., Lin, C.-J., and Weng, R. C. (2004). Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research*, 5:975–1005.
- [298] Yamshchikov, I., Tikhonov, A., Pantis, Y., Schubert, C., and Jost, J. (2022). BERT in Plutarch’s shadows. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6071–6080. Association for Computational Linguistics.
- [299] Yang, Y. and Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning (ICML 1997)*, pages 412–420, Nashville, US.
- [300] Young, T., Hazarika, D., Poria, S., and Cambria, E. (2018). Recent trends in Deep Learning based Natural Language Processing. *IEEE Computational Intelligence Magazine*, 13(3):55–75.
- [301] Yu, L., Zhang, W., Wang, J., and Yu, Y. (2017). SeqGAN: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.
- [302] Yule, G. U. (1939). On sentence-length as a statistical characteristic of style in prose: With application to two cases of disputed authorship. *Biometrika*, 30(3/4):363–390.
- [303] Zadrozny, B. and Elkan, C. (2002). Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the 8th ACM International Conference on Knowledge Discovery and Data Mining (KDD 2002)*, pages 694–699, Edmonton, CA.
- [304] Zhai, W., Rusert, J., Shafiq, Z., and Srinivasan, P. (2022). A girl has a name, and it’s... Adversarial authorship attribution for deobfuscation. *arXiv preprint arXiv:2203.11849*.
- [305] Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems (NIPS)*, volume 1, pages 649–657, Cambridge, MA, USA. MIT Press.
- [306] Zhang, Y., Gan, Z., Fan, K., Chen, Z., Hénao, R., Shen, D., and Carin, L. (2017). Adversarial feature matching for text generation. In *International Conference on Machine Learning*, pages 4006–4015. PMLR.
- [307] Zheng, R., Li, J., Chen, H., and Huang, Z. (2006). A framework for authorship identification of online messages: Writing-style features and classification techniques. *Journal of the American society for information science and technology*, 57(3):378–393.
- [308] Zugarini, A., Melacci, S., and Maggini, M. (2019). Neural poetry: Learning to generate poems using syllables. In *International Conference on Artificial Neural Networks*, pages 313–325. Springer.

## Secondary sources

- [309] Alessio, G. C. (1983). *Bene Florentini Candelabrum*. Editrice Antenore, Padova, IT. <https://bit.ly/2Xb10pR> (Archivio della Latinità Italiana del Medioevo), accessed 2018-05-28.
- [310] Auzzas, G. (1992). *Tutte le opere di Giovanni Boccaccio*, chapter “Epistole e lettere”. Mondadori, Milano, IT. <https://bit.ly/2I2VLjp> (Biblioteca Italiana), accessed 2018-05-28.
- [311] Azzetta, L. (2016). *Nuova edizione commentata delle opere di Dante*, volume 5, chapter “Epistola XIII”, pages 271–487. Salerno Editrice, Roma, IT.
- [312] Baldeschi, L. C. (1925-1926). Le “Constitutiones Romandiolae” di Giovanni d’Appia. *Nuovi Studi Medievali*, 2(1):221–252. <https://bit.ly/30RRoAg> (Archivio della Latinità Italiana del Medioevo), accessed 2018-05-28.
- [313] Brownlee, K. and Hollander, R. (2018). Benevenuti de Rambaldis de Imola Comentum super Dantis Aldigherij Comoediam, nunc primum integre in lucem editum sumptibus Guilielmi Warren Vernon, curante Jacobo Philippo Lacaita. Florentiae, G. Barbèra, 1887. <https://bit.ly/2Dqj6du> (Darmouth Dante Project), accessed 2018-05-28.
- [314] Chiamenti, M. (1999). *I commenti danteschi dei secoli XIV, XV e XVI*, chapter “Comentum super Comedie Dantis (terza ed ultima redazione del “Comentum”)”. LEXIS Progetti Editoriali, Roma, IT. <https://bit.ly/2ECcU2c> (Biblioteca Italiana), accessed 2018-05-28.
- [315] Cioffari, V. (1974). *Guido da Pisa’s Expositiones et Glose super Comediam Dantis, or Commentary on Dante’s Inferno*. State University of New York Press, Albany, US. <https://bit.ly/2FRhT0x> (Darmouth Dante Project), accessed 2018-05-28.
- [316] Clark, E. (1997). Magistri Boncompagni Ysagoge. *Quadrivium*, (8):23–71. <https://bit.ly/2MedBFc> (Archivio della Latinità Italiana del Medioevo), accessed 2018-05-28.
- [317] Cotza, V. (2013). Giovanni del Virgilio, Allegorie super fabulas Ovidii Methamorphoseos. Edizione critica e introduzione. Master’s thesis, Department of Philology, Literature and Linguistics, University of Pisa, Pisa, IT.
- [318] Daffinà, P., Leonardi, C., Lungarotti, M. C., Menestò, E., and Petech, L. (1989). *Giovanni di Pian di Carpine. Storia dei Mongoli*, pages 227–333. Fondazione CISAM, Spoleto, IT. <https://bit.ly/2WoZaSM> (Archivio della Latinità Italiana del Medioevo), accessed 2018-05-28.
- [319] D’Angelo, E. (2014). *L’epistolario di Pier della Vigna*. Rubbettino Editore, Soveria Mannelli, IT.
- [320] Fabbri, R. (1992). *Tutte le opere di Giovanni Boccaccio*, chapter “De vita et moribus d. Francisci Petracchi”. Mondadori, Milano, IT. <https://bit.ly/2MdBUTV> (Biblioteca Italiana), accessed 2018-05-28.
- [321] Ferrario, F. (1999). *Expositio seu comentum super “Comedia” Dantis Allegherii*, a cura di Saverio Bellomo. Florence: Le Lettere, 1989. <https://bit.ly/2Uac1mS> (Darmouth Dante Project), accessed 2018-05-28.
- [322] Frugoni, A. and Brugnoli, G. (1996). *Dante Alighieri - Opere minori*, chapter “Epistole”. Riccardo Ricciardi Editore, Milano, IT. <https://bit.ly/2JIPYTp> (Biblioteca Italiana), accessed 2018-05-28.

- [323] Garbini, P. (1996). *Boncompagnus de Signa - Rota Veneris*. Salerno Editrice, Roma, IT. <https://bit.ly/2wrCTVS> (Archivio della Latinità Italiana del Medioevo), accessed 2018-05-28.
- [324] Garbini, P. (1999). *Boncompagnus de Signa - Liber de obsidione Ancone*. Viella, Roma, IT. <https://bit.ly/2QuMMLw> (Archivio della Latinità Italiana del Medioevo), accessed 2018-05-28.
- [325] Garufi, C. A. (1937). Ryccardi de Sancto Germano notarii Chronica. *Rerum Italicarum Scriptores*, 7(2). <https://bit.ly/2HHrE1W> (Archivio della Latinità Italiana del Medioevo), accessed 2018-05-28.
- [326] Gaudenzi, A. (1971a). *Guido Faba - Dictamina Rhetorica Epistole*, chapter “Guidonis Fabe Epistole”. Forni Editore, Bologna, IT. <https://bit.ly/2WrxgWh> (Archivio della Latinità Italiana del Medioevo), accessed 2018-05-28.
- [327] Gaudenzi, A. (1971b). *Guido Faba - Dictamina Rhetorica Epistole*, chapter “Guidonis Fabe Dictamina Rhetorica”. Forni Editore, Bologna, IT. <https://bit.ly/2I36vhI> (Archivio della Latinità Italiana del Medioevo), accessed 2018-05-28.
- [328] Griffin, N. E. (1936). *Guido De Columnis - Historia destructionis Troiae*. The Mediaeval Academy of America, Cambridge, US. <https://bit.ly/2EF02IR> (Archivio della Latinità Italiana del Medioevo), accessed 2018-05-28.
- [329] Kristeller, P. O. (1961). Un’ “Ars dictaminis di Giovanni del Virgilio”. *Italia Medioevale e Umanistica*, (4):179–200. <https://bit.ly/2MeCD7k> (Archivio della Latinità Italiana del Medioevo), accessed 2018-05-28.
- [330] Mancuso, S. (2015). Benvenuto da Imola, *Glose Bucolicorum Virgilii* (ad Buc. I, IV, VI, X). Studi critici ed edizione. Master’s thesis, Department of Philology, Literature and Linguistics, University of Pisa, Pisa, IT.
- [331] Meloni, P. (1953). *Nicolai Treveti Expositio L. Annaei Senecae Agamemnonis*. Centro di Studi Filologici e Linguistici Siciliani, Palermo, IT. <https://bit.ly/2KfRxYt> (Biblioteca Italiana), accessed 2018-05-28.
- [332] Menestò, E. and Brufani, S. (1995). *Fontes Franciscani*. Edizioni Porziuncola, Assisi, IT.
- [333] Monleone, G. (1941). *Jacopo da Varagine e la sua Cronaca di Genova dalle origini al MCCXCVII*, pages 3–414. FSI, Roma, IT. <https://bit.ly/2MhK1xs> (Archivio della Latinità Italiana del Medioevo), accessed 2018-05-28.
- [334] Nardi, B. (1996). *Dante Alighieri - Opere minori*, chapter “Monarchia”. Riccardo Ricciardi Editore, Milano, IT. <https://bit.ly/2EEGQLg> (Biblioteca Italiana), accessed 2018-05-28.
- [335] Romano, M. M. M. (2004). *Corpus Christianorum Continuatio Mediaevalis CLXXXIII*, chapter “Raimundus Lullus – Ars amativa boni”, pages 120–432. Brepols Publisher, Turnhout, BE. <https://bit.ly/2wnBu2t> (Archivio della Latinità Italiana del Medioevo), accessed 2018-05-28.
- [336] Romano, V. (1951). *Giovanni Boccaccio - Genealogie deorum gentilium libri*. Laterza, Bari, IT. <https://bit.ly/2ZEvcZI> (Biblioteca Italiana), accessed 2018-05-28.
- [337] Rossi, L. C. (1999). *I commenti danteschi dei secoli XIV, XV e XVI*, chapter “Commento all’Inferno di Dante”. LEXIS Progetti Editoriali, Roma, IT. <https://bit.ly/2EzazoX> (Biblioteca Italiana), accessed 2018-05-28.
- [338] Rossi, L. C. (2002). “Beneventus de Ymola super Valerio Maximo”. *Ricerca sull’Expositio. Aevum - Rassegna di Scienze Storiche Linguistiche e Filologiche*, (76):369–423.

- [339] Schneider, F. (1926). Untersuchungen zur italienischen Verfassungsgeschichte: Staufisches aus der Formelsammlung des Petrus de Boateriis. *Quellen und Forschungen aus italienischen Archiven und Bibliotheken*, (18):191–273.
- [340] Stok, F. (1991). La “Vita di Virgilio” di Zono de’ Magnalis. *Rivista di Cultura Classica e Medioevale*, 33(2):143–181.
- [341] Sutter, C. (1894). *Aus Leben und Schriften des Magisters Boncompagno*. Akademische Verlagsbuchhandlung von J.C.B. Mohr, Freiburg im Breisgau, DE. <https://bit.ly/2ws07eg> (Archivio della Latinità Italiana del Medioevo), accessed 2018-05-28.
- [342] Tavoni, M. (2011). *Dante Alighieri - Opere*, chapter “De vulgari eloquentia”. Mondadori, Milano, IT. <https://bit.ly/2HI1fBRW> (DanteSearch), accessed 2018-05-28.
- [343] Ussani, Jr, V. (1959). *L. Annaei Senecae Hercules furens et Nicolai Treveti expositio*. Edizioni dell’Ateneo, Roma, IT. <https://bit.ly/2KccRxN> (Biblioteca Italiana), accessed 2018-05-28.
- [344] Zaccaria, V. (1967). *Tutte le opere di Giovanni Boccaccio*, chapter “De mulieribus claris”. Mondadori, Milano, IT. <https://bit.ly/2I1VWv1> (Biblioteca Italiana), accessed 2018-05-28.