UNIVERSITY OF CRETE

Department of Philology

Postgraduate Study Programme: Master of Arts (MA) in Linguistics

A Computational Criticism Analysis of Papadiamantis' Work:

Language, Topics, and Clustering

Author: Dimitrios Papadakis (Student ID: 0961)

Supervisor: Stergios Chatzikyriakidis

Rethymno, Greece

September 2025

Contents

Acknowledgments

Στη Γαλάτεια

There were many people responsible for the completion of this thesis and to whom I owe my gratitude. However, since the reasons I wish to thank them for could fill a thesis in themselves, I shall confine myself to mentioning their names, which I shall accompany with explanations where I feel it necessary.

Firstly, I would like to thank my family, my sister Anna and my parents, Linda and Michalis, for their continuous support throughout my studies.

I would also like to thank my beloved fellow students and friends from the postgraduate programme in Crete. Thank you to Thodoris, Danae, Konstantinos, Manuela, Ilias, Christina, Efthimia, Konstantinos, Dimitris, Kleanthi, Manolis and Irini. I would like to give special thanks to Angelos, Chara, Erofili, Vasiliki, Irini for their continuous support in this thesis, their very useful advice, without whose help and love I would not have been able to complete this work.

Among the people who supported me on this journey are people outside the postgraduate programme, and their love and support have always been particularly important to me throughout this journey, and I appreciate it all very much. I would like to thank Paraschos, Kostas, Giorgos, Kostas, Dimitris, Angeliki, Dia, Irini, Manos and Meletis by name.

Next, I would like to thank both the professors of the Linguistics Department of the Faculty of Philology at the University of Crete and those of the Modern Greek Literature Department. I would especially like to thank Professor Ioanna Kappa, my academic supervisor, for her helpful and supportive guidance, but also for the opportunity to get involved in research in the Variphon program.

I would now especially appreciate the opportunity to thank Prof. Anastasia Natsina for the inspiration I gained from her undergraduate courses and for her valuable advice on this thesis, effectively acting as my informal co-supervisor, Prof. Marina Aretaki, for her help in supporting my interpretative approach and helping me to explore the world of Papadiamantis through her

inspired critique of his work, Prof. Dimitris Polychronaki for giving me the opportunity to try out the application of computational tools for literary criticism in his postgraduate seminar entitled *'The Literature of Laughter'*, Prof. Despoina Oikonomou, for her inspiring semantics and pragmatics classes and her very positive attitude to the Master's program, Prof. Elena Anagnostopoulou, for her inspiring courses on linguistic typology, Prof. Vina Tsakali, for giving us the opportunity to participate in the international summer school of linguistics in Rethymno, Dr. Maria Barouni, for inspiring me to study linguistics, for guiding me through my first undergraduate linguistics seminar, providing me with the resources for the diachronic study of Greek, through the lens of the interface between syntax and semantics, and for her unforgettable presence at the University of Crete, as a teacher and researcher, Prof. Georgia Katsimali, because her classes are my first memory of the world of linguistics, and finally, Prof. Alexis Kalokairinos, because he convinced us very early on that it is worthwhile to question and doubt, both as linguists, and as scientists. Alongside my professors, I would like to thank Mr. George Motakis for his excellent administrative support and guidance, his good humor, and his thoughtful academic advice.

I would also like to thank Prof. Maxime Amblard, from LORIA in Nancy, France, where I did my internship, for giving me the opportunity to work on computational embedding models and see how they work first-hand. In particular, I would like to thank the people who became like family to me in Nancy: Dimitris, Argyris, Kostas, Dionysis, Dimitris, Dimitra, Stella, Yannis and Ilias (yes, there were a lot of Greeks-Ναντσιώτες), but also my friends from the research lab where I did my internship, Remy, Hee-soo, Siyana, Amandine and all doctoral students, professors, and researchers of the Semagramme team.

Furthermore, for my stay in Skiathos last summer, in the birthplace of Alexandros Papadiamantis, I would like to thank Maria and Eleni Gkonta for their hospitality and kindness.

Last but not least, I would like to thank my thesis supervisor, Professor Stergios Chatzikyriakidis, whose example as a researcher, educator, and person has been a constant source of inspiration at the University of Crete. Prof. Chatzikyriakidis is a teacher in every sense of the word, supportive, respectful of each student's freedom, building bridges for all of us, with open-mindedness, insightful judgement, monster knowledge of the scientific field, simplicity, disarming humour, generosity, kindness and goodness make him a person you really enjoy working with and having as a professor. Fortunately, I met Prof. Chatzikyriakidis when I was still an undergraduate student and had the opportunity to collaborate on several projects as a

member of his teams. Although I mentioned in my interview, when I entered the postgraduate programme, that I would like to combine computational linguistics and literature in my research, I never imagined that the journey would be so fun and educational for me working with him. 'Κύριε Χάτζη, σας ευχαριστώ! Έχω ηρεμήσει!'

Abstract

This thesis explores Alexandros Papadiamantis' work through the lens of computational criticism, employing natural language processing (NLP) and machine learning (ML) to address three key research questions. First, it tackles the task of language identification in Papadiamantis' texts by training a Multinomial Naive Bayes classifier on a custom dataset of 10 million words representing Standard Modern Greek, Katharevousa, and Ancient Greek. Additionally, narrative and dialogic parts of the texts were distinguished using textual patterns that introduce dialogue, allowing for a detailed analysis of language use across different parts of the corpus. Second, it identifies prominent topics within the author's short stories using word embedding models (Word2Vec and FastText) combined with clustering algorithms, with optimal hyperparameters identified through grid search. Third, it applies contextual embeddings (BERT) and clustering techniques to categorize Papadiamantis' works based on semantic similarity, again using grid search to select the best-performing configurations. The analysis reveals that Katharevousa predominates across genres, but that Modern Greek is more prevalent in dialogues, especially in short stories, without, in any case, exceeding the percentage of use of the Katharevousa. Thematic clusters highlight key motifs such as Christian liturgical language, female roles, socio-economic structures, etc., offering new insights into Papadiamantis' literary landscape. Furthermore, the clustering of all Papadiamantis' works reveals semantic groupings that transcend traditional chronological or genre-based divisions, offering a fresh, data-driven perspective on his corpus. This study demonstrates how computational methods can complement and extend traditional literary criticism by providing a new, integrated quantitative and qualitative approach to Papadiamantis studies and contributing to the broader analysis of Greek literary texts.

Chapter 1

Introduction

In this master's thesis, we will attempt to explore three central issues in Papadiamantis studies which, to the best of our knowledge, have not yet been examined using computational methods. First, what percentage of Modern Greek, Katharevousa, and Ancient Greek are used in the works of Alexandros Papadiamantis, and how does their use differ across narrative and dialogue, in both novels and short stories? Secondly, what are the prominent topics identified in Papadiamantis' short stories? And thirdly, how can the author's short stories and novels be clustered in terms of semantic similarity in the multidimensional semantic space?

In order to investigate the above, this thesis is structured into the following chapters. Chapter 2 presents the framework of this thesis, describing and proposing computational criticism as a method for studying literature, while also discussing how it differs from traditional literary criticism. Chapter 3 describes the corpus of works by Alexandros Papadiamantis that was used for our analysis. Continuing, Chapter 4 investigates the various linguistic varieties used by Papadiamantis through the task of language identification, while Chapter 5 seeks to identify the main thematic axes in the author's short stories through embedding models and clustering algorithms. Finally, in Chapter 6, the short stories and novels are categorised using a version of the BERT model for Greek together with a clustering algorithm. In both Chapter 5 and Chapter 6, the search for the optimal models and their parameters is carried out using a grid-search method. The conclusions and bibliography follow.

Each chapter is structured as follows: It begins with an introductory section, followed by indicative relevant work on the topic under investigation [1] and focuses on analysis, after

[1] Not in the field of computational analysis of Papadiamantis, as there is no such work apart from that of Mikros (2020) which focuses only on his translation work, performing the authorship attribution task, i.e. attempting to certify the author.

describing basic concepts and terms [2]. Each chapter concludes with a presentation of the results, followed by a discussion. In addition, Chapter 4 includes a manual error analysis, while Chapters 5 and 6 restrict evaluation to clustering metrics (Cohesion, Silhouette, DBCV; Davies–Bouldin). The chapters were inspired by the three questions in Papadiamantian studies, while an attempt was made to find suitable computational tasks that can explore them.

More broadly, an attempt was made to establish a connection between the existing extensive bibliography of Papadiamantis studies and computational methods using NLP. However, we are aware that this work follows a different path from traditional literary analysis and therefore employs different methods of drawing concludes than those that would be used in a traditional analysis. The modelled method, as proposed in the theoretical background, seems to be able to bring computational methods and literary criticism analysis together within a qualitative and quantitative research framework, offering new interpretations for literary analysis, in negotiation with previous views, providing a more positive approach to the study of literature than the existing one. In this context, this thesis approaches the three research questions through the lens of a combination of qualitative and quantitative analysis of Papadiamantis' works, answering the three research questions in the here and now of models, datasets and within the framework of a master's thesis[3].

[2]Key terms and concepts introduced in chapter x are also used in chapter x+1.

[3]The full code and results presented in this thesis are available here: https://github.com/dimitrispapad/Papadiamantis

Chapter 2

Theoretical Background- Computational Criticism

This master thesis is framed within an approach that adopts the view that data and computation play a significant role in the way we interpret literature. However, as Piper (2019) mentions, we still do not have a completely clear picture of how quantitative methods respond to specific questions within the discipline of literary studies. In his book-Enumerations- which inspired this work, Piper (2019) attempts to offer methods that have much to contribute to the field of computational criticism. He specifically notes that two fundamental aspects of the discipline have been overlooked. The first is a well-preserved textual repetitiveness. Every text consists of elements that repeat frequently. These repetitions, in turn, perpetuate around writing, across genres, periods, themes, etc., while we choose to focus on rare writing events, having no accurate way to calculate repetitiveness. The second is the problem of generalization. Generalization is critical in any scientific methodology. It is the way we move from the part to the whole. For example, until recently, we had no way to test our hypotheses on a very large collection of novels, or on specific parameters that can be expressed in quantitative terms, thus providing arguments for '*the novel*'. Meanwhile, Piper (2019) continues, we spend a lot of time training readers and researchers to be more attentive to what is essentially in the text, we spend much less time on the process of generalization itself. Thus, the way that will help us generalize, which is absent from the humanities studies, is modeling.

But what are models and how can we use them in the study of literature? Models are primarily what mediate between us and our observations. By focusing on models, we think small, so we can think big, moving from the binary logic of size to the logic of representations.

Thus, we begin to reflect on the representativeness of our evidence, large and small, close and distant, becoming interwoven. Computational reading is far more circular than has so far been acknowledged. Whereas the social sciences often speak in terms of '*samples*' and '*bias*', the notion of '*representativeness*' suggests that there is not ultimately some stable, knowable whole against which one must limit one's '*bias*'. The following figure presents the modeling proposed by Piper (2019, p. 10) in literature.

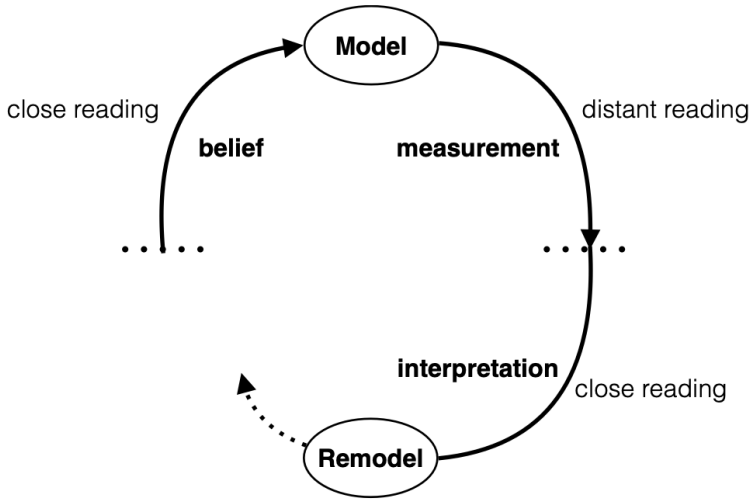


Figure 2.1: Diagram illustrating the approach to modeling in literary studies, as discussed by Piper (2019, p. 10)

Comparing this kind of literary modeling to traditional literary criticism, what is different in this approach is that there is the possibility of comparing a '*part*' and a '*representation*' of the whole. In traditional literary criticism, the hypothesis is formulated, and it matches the parts we select. We rarely hear about works that do not match our hypothesis. In computational literary criticism, the hypothesis is set and tested on a very large set of documents. Moreover, this modeling makes the study of literature '*more architectonic and less agnostic, more social and more collective*' (Piper, 2019), as the methods, the research trajectory, and the results are always available for evaluation. Therefore, it is not something true once and for all, but an interpretation of the texts under examination, shaped by the lens of this criticism and by the specific conditions under which the models were run. Since the models are dynamic, their application necessarily involves change. After all, Piper (2019) concludes that models models are ongoing.

This concept of computational criticism of literature, of course, did not emerge in a vacuum, but is closely linked to the development of the Digital Humanities (DH) and Natural language processing (NLP), that is the tendency of the humanities to engage the possibilities of computer

science and the processing of large amounts of data, but also the direction of computer science to process natural language. In fact, the founder of Digital Humanities is Roberto Busa, an Italian Jesuit priest, who in the late 1940s initiated the production of an automatically generated concordance to the works of Thomas Aquinas using a computer (Busa, 1980; Ramsay, 2011)[1]. From that first work until the next sixty years, the generation of such transformations of the text for humanistic purposes was associated with the term '*text analysis*', while in literature in particular this computational text analysis was associated with problems of style and author identification. Then, with the emergence of NLP in the 1990s and from the early 2000s (Hirschberg & Manning, 2015) (but also the tremendous growth that it led to today's Large Language Models (LLMs)), the field of DH was broadened with its incorporation and also with the addition of artificial intelligence, statistical computing, corpus linguistics, and data mining (Ramsay, 2011).

Roseanne Potter, as cited in Ramsay (2011), argued in the late 1980s that linking literature and technology was valuable because the principled use of technology and criticism required that criticism become fully comfortable with the scientific method. Even today, some literature professors often follow what John Ellis called in 1974, as noted in Ramsay (2011), wise eclecticism, namely the idea that if an argument is well made, it is just as valid as any other. While literary criticism has produced many rich interpretations, most of them remain hypotheses. They are rarely tested against large amounts of data and are usually supported only by selective examples. This is where computational methods, including data computation (Ramsay, 2011) and modeling (Piper, 2019), become especially important.

Then, we can conclude by stating that computational methods can help the study of literature in many ways, centred on modelling its study, testing hypotheses on a large amount of data, using advanced NLP techniques for analysis, comparative work, but only as long as criticism is open to this versatile approach. Of course, computational criticism cannot solve all problems, nor does it function in an automatic way that yields correct interpretations. The research questions are determined by our choices; the data and text sets are constructed by us; the models are run according to the parameters we define; and the interpretations of the results are again carried out by us. Ultimately, computational literary criticism seeks to foreground the interpretive approach to texts, highlighting the relationships between texts and interpretations, between models and

[1]With support from IBM, Busa's work marked a foundational moment in the field: the creation of a radically transformed, reordered, and searchable version of one of the world's most influential philosophical corpora (Schreibman et al., 2008).

interpretations, and between research questions and interpretive methods. The objectivity claimed by this approach rests on the transparency of the interpretive process itself, while still acknowledging that subjectivity remains an inherent element of this scientific endeavor.

From this perspective, we may argue that we are adopting a different way of thinking from that proposed by Chatzivasileiou (2001), one that assumes that the interpretation of Alexandros Papadiamantis' work is inevitably defined by the weight of prior critical readings. While acknowledging their influence, we suggest that a contemporary analysis can, to some extent, operate with a degree of independence. In fact, although it is possible to explore diachronic research questions within Papadiamantian studies, our primary aim is to offer a renewed interpretation, to reread Papadiamantis' texts, placing data at the center, as our analytical foundation and a new way of analyzing the literature text. Through this analysis, we can therefore see the relationship between interpretations to the present day and those produced within the framework of computational literary criticism, while Papadiamantis' corpus is offered for this investigation due to its size and availability.

Chapter 3

Alexandros Papadiamanti's Corpus

3.1 Data Collection and data processing

As Piper (2019) notes, the notion of a writer's '*corpus*' has, since Cicero, symbolized the author's body of work, a collection that is at once cohesive and complete, finite and bounded. The corpus functions as a tangible counterpart to the author's life, imparting shape and meaning to its otherwise inchoate aspects. This dual sense of completeness and vulnerability defines the corpus both textually and biologically, marking its beginnings and endings. In computational literary criticism, the corpus used each time is a methodological decision, in the sense that, in combination with the models used, it frames the analysis and is closely linked to the results that are produced.

As we know, Papadiamantis's entire body of work was published after his death in the edition carried out by Triantafyllopoulos (Papadiamantis, 1981–1988). The part that concerns us is his prose works, specifically his short stories and novels, while during his lifetime his short stories (i.e., the greater part of his work) were not published collectively.

For the current study, the corpus of Alexandros Papadiamantis was compiled from two distinct sources, resulting in two separate versions, each suited to the nature of specific tasks. The first version comes from the digital form provided by Dimitroulia (2021). The second version was manually assembled in text files from the online collection available through the Association for Papadiamantis Studies [1]. Both versions encompass the same body of work, specifically including novels, short stories, and poems, following the editions of Papadiamantis (1981–1988) by Triantafyllopoulos. They differ however in terms of how they were collected, as

[1] https://papadiamantis.net/

one version contains all the works of each genre consolidated into a single file, while the other was collected manually, with each work in a separate file, which explains the slight difference in the number of words.

Both versions were converted from polytonic to monotonic format to streamline processing and ensure consistency with Modern Greek linguistic conventions [2]. Thus, the final corpus exists in two forms: the CLARIN version, referenced as the standard corpus, and the manually assembled version, created to facilitate specific tasks, with the material arranged in chronological order, since each work of Alexandros Papadiamantis is contained in a single text file. Each task includes a statement indicating which of the two corpora is used. The table below shows the two corpora in detail in terms of word count and categories:

Category	**CLARIN Corpus (words)**	**Manual Corpus (words)**
Total	665,820	649,836
Short Stories	466,977	458,124
Novels	195,720	188,670
Poems	3,122	3,042

Table 3.1: Word count comparison between the CLARIN corpus and the manually assembled corpus from the Association for Papadiamantis Studies.

[2]We converted the text from polytonic to monotonic Greek using the tool available at https://neobabis.gr/portfolio/politoniko-se-monotoniko/

Chapter 4

Language Identification in Papadiamantis' Texts

4.1 Introduction

The language of Alexandros Papadiamantis has long been a focal point of scholarly debate, defining his literary identity and setting him apart from his contemporaries (Φαρίνου-Μαλαματάρη, 2005). According to the literature, his language resists singular categorization (Τωμαδάκης, 2005), resulting in interpretations that are often diverse and contradictory (Ραγκαβής, 1908; Ψυχάρης, 1905, among else). Researchers have pointed out the complex and varied nature of his language usage, which blends elements from several varieties of Greek (Άγρας, 1934; Τωμαδάκης, 2005, amanong else). This chapter draws inspiration from the comprehensive work of Φαρίνου-Μαλαματάρη (2005), whose critical anthology provides a significant foundation for understanding the diverse scholarly perspectives on Papadiamantis' language [1].

Here, we explore a central problem in Papadiamantic studies: In what language does Alexandros Papadiamantis write his short stories and novels, and how much language is differentiated between dialogical and narrative parts? In response to the controversy in the Papadiamantic studies literature, we attempt through classical Machine Learning algorithms such as Naive Bayes to measure the percentage of each of the following linguistic varieties used by Papadiamantis: Ancient Greek, Katharevousa, Modern Greek. In order to accomplish this Language

[1]While the discussion here integrates direct references to other scholars, it acknowledges Φαρίνου-Μαλαματάρη (2005) as a critical resource for collating and interpreting these varied viewpoints. Such an approach ensures that the chapter remains anchored in the broader literary discourse while contributing a computational perspective to the ongoing conversation.

Identification task, we construct a dataset consisting of approximately 10,000,000 words derived from each language variety. With this dataset we train our models to detect each of the three language varieties. Then, after distinguishing the dialogical from the narrative parts, we ask the best-performing algorithm to classify the data from the short stories and those from the novels into each of the three categories.

4.2 Related Work

As Λορεντζάτος (1994) notes, Papadiamantis employs the entirety of the Greek language, drawing from its historical breadth, from its earliest forms to his own era. Tziovas (1989) suggests that his language merges orality and textuality, while Φαρίνου-Μαλαματάρη (2005) argues that language functions as the primary organizing principle of his narrative structure, even to the extent of overturning the primacy of the narrative structure itself.

Supporters of Katharevousa, such as Ραγκαβής (1908), criticized Papadiamantis for using language as a '*rustic peasant girl*', while even supporters of Demotic Greek (i.e. the Standard Modern Greek) like Βλαστός (1911) accused him of technical disorder and negligence in structure-likely because he did not exclusively use Demotic Greek. Even Ψυχάρης (1905), in his 1905 list of Demoticist writers, excluded Papadiamantis, claiming that '*if you remove a few phrases in Demotic from Papadiamantis' language, it resembles Katharevousa much more*'.

The debate around Papadiamantis' language extends beyond the dichotomy of Katharevousa versus Demotic. In the literature, one finds a division of opinion. Τερζάκης (1937) describes his language as problematic, while Παπαϊωάννου (2005) sees it as a survival of the past, deeply ingrained in his linguistic psyche despite his respect for the people's language. Μουλλας (1974) perceives it as rebelliously adorned in the garb of Katharevousa. Others, like Άγρας (1934), see it as a language enriched with layers of education and history, borrowing from all linguistic periods, refusing to conform to the monotony or purity of any single category. Τωμαδάκης (2005) highlights the diachronic layers of language that reflect an unyielding resistance to any specific classification, while they emphasize the linguistic hybridity inherent in Papadiamantis' style.

On a broader level, those who view Papadiamantis' language positively admire its linguistic diversity (including elements of Archaising, Katharevousa, Demotic, ecclesiastical language, and journalistic style). According to Βαλέτας (1941), it is a language that successfully conveys

the popular element to the urban class, dressed in the appropriate attire required by print culture. Another favorable interpretation, as Φαρίνου-Μαλαματάρη (2005) notes citing Παπαγιώργης (1997), is that it delivers the familiar in an unfamiliar form, closely aligning with the Russian Formalist notion of defamiliarization, through linguistic stratification that reflects the varied experiences of seemingly simple characters. It is no coincidence that the Surrealists, who championed both Demotic and Katharevousa, appreciated Papadiamantis' language. Εγγονόπουλος (1999) praised it, an opinion later reiterated by Ελύτης (1997), who described Papadiamantis' language as combining synchronic and diachronic elements, reinforcing his argument for Papadiamantis' typological depiction of Greek life. More recent interpretations describe his language as producing a modern style that merges the primitive with the cultivated, the oral with the linguistically elaborate, an unusual, authentic, magical mix that is simultaneously naive and sophisticated (Πασχάλης, 2001).

Although often revealing about the artistic effect of Papadiamantis' work, the aforementioned perspectives on Papadiamantis' language are not derived from a purely linguistic analysis. Instead, they are deeply influenced by the ideological and aesthetic positions of the scholars regarding linguistic matters. This observation reveals a gap in the literature, which this study aims to address through a computational examination of Papadiamantis' language. It is crucial to clarify that this investigation does not stem from any ideological or aesthetic stance. Instead, the task of Language Identification employs Machine Learning techniques to train models capable of recognizing and categorizing language. Specifically, these models will classify the text into one of three linguistic categories central to this discussion: Ancient Greek, Modern Greek, or Katharevousa. The study deliberately avoids broader social categorizations, such as '*urban language*' or '*language of people*', as defining such categories would introduce significant complexity.

The working hypothesis for this analysis builds on prior observations (including those by Τωμαδάκης (2005) and Ψυχάρης (1905, among else)): the narrative sections of Papadiamantis' texts are hypothesized to resemble Katharevousa more closely, while the dialogic sections are hypothesized to align more with Modern Greek. This hypothesis reflects the linguistic duality often noted in Papadiamantis' work. The third class, Ancient Greek, is based on the argument that there are elements of Ancient Greek in the Papadiamantis corpus(Νάκας, 2003; Παπαγιώργης, 1997, among else), and because of its close relationship with Katharevousa. Thus, the specific research question at the center of this study is as follows: To what extent can

the language used in Papadiamantis' dialogic and narrative sections be classified into each of these three linguistic categories? This computational analysis seeks to provide a systematic and data-driven exploration of Papadiamantis' linguistic style, shedding new light on its classification and bridging the gap between traditional interpretive approaches and computational criticism methodologies [2].

4.3 The Linguistic Landscape: Ancient Greek, Modern Greek, and Katharevousa

This section explores the linguistic landscape pertinent to Alexandros Papadiamantis' work, focusing on three linguistic varieties that scholars have associated with his writings: Standard Modern Greek (Demotic), Katharevousa and Ancient Greek. A priori, it is important to note that contemporary linguistic research adopts the following division of Greek language evolution into historical periods (Horrocks, 2014) as shown in Table 4.1 [3]:

Period	Timeframe
Ancient Greek	1400–300 BCE
Hellenistic Koine	300 BCE–6th century CE
Medieval Greek	6th–18th century CE
Modern Greek	19th–20th century CE

Table 4.1: Chronological division of Greek language evolution (Horrocks, 2014).

We decided to examine only these three linguistic varieties, conscientiously leaving out the Skiathos dialect (and any dialectal variety in general), as its influence on his work will not be addressed in this study, due to a lack of data from the dialectal variety of Skiathos. But let us first make a brief reference to each of the linguistic varieties we are dealing with.

4.3.1 Standard Modern Greek (SMG)

Standard Modern Greek (SMG) is the official language of the Greek state, derives from Koine Greek, as most Modern Greek dialects (except Tsakonian) [4]. It is spoken by approximately 13

[2]To answer this question, we use all of Alexandros Papadiamantis's work, except for his poems. The corpus of Papadiamantis's work used here comes from the digital form provided by Dimitroulia (2021), based on the edition of Papadiamantis's complete works by Triantafyllopoulos (Papadiamantis, 1981–1988), and includes the following: Short Stories (466,977 words) and Novels (195,720 words)

[3]Katharevousa does not appear in the table 4.1 as it is not an evolutionary stage of the Greek language

[4]Standard Modern Greek (SMG) is recognized as the official language of both Greece and Cyprus. Its linguistic foundations are rooted in the Peloponnesian Greek dialects, as noted by Mackridge (1985). The emergence of

million people, primarily in Greece, with around 10 million speakers, and in Cyprus, home to an estimated 500,000 speakers. Additionally, significant Greek-speaking communities exist in the diaspora, notably in Melbourne, Australia, which houses around 1 million speakers (Moschonas, 2016).

4.3.2 Ancient Greek (AG)

Contrary to the notion of a single unified language, Ancient Greek (AG) consisted of numerous dialects that evolved over centuries across different regions of the Greek mainland and the coastal areas of Asia Minor (Horrocks, 2014).

When discussing AG today, it is often understood to refer specifically to the Attic dialect, the language of Athens during the 5th century BCE. This dialect was associated with the intellectual and cultural achievements of Classical Greece, including tragedy, philosophy, rhetoric, historiography, and other sciences. This admiration for the intellectual contributions of the Athenians has fostered a similarly high regard for their language.

However, historical linguistics reveals that AG encompassed far more than just Attic Greek (Horrocks, 2014). For instance, earlier forms such as Homeric Greek (from the 8th century BCE) and later forms like Koine Greek (circa 300 BCE–6th century CE) all fall within the broader category of AG.

For the purposes of this study, the AG dataset includes texts spanning from the 8th century BCE to the 1st century CE. This dataset ensures coverage across various genres, reflecting the richness and diversity of AG language use across historical periods and literary forms. These include works from various genres, as shown in table 4.2:

4.3.3 Katharevousa

Katharevousa, described as a '*purist*' variety of SMG, served as the official written language of Greece until 1976. It was employed in government and judiciary documents, as well as in most newspapers and technical publications. Emerging in the 19th century, Katharevousa was part of an effort to '*purify*' the Greek language by eliminating foreign elements and systematizing its

SMG can be traced back to the Greek War of Independence (1821–1829), during which the Peloponnese was one of the first regions to be liberated. Additionally, the dialects of this region, with the exception of Tsakonian, closely aligned with the written Greek language of the time. This standard form of the language was further influenced by linguistic elements from the dominant Greek communities of the period, such as those in Istanbul and the Ionian Islands (Chatzikyriakidis et al., 2023).

Genre	Example Texts
Rhetoric	Aeschines' *Against Timarchus* (4th century BCE)
Historiography	Thucydides' *The Peloponnesian War* (5th century BCE)
Philosophy	Epictetus' *Enchiridion* (1st century CE)
Epic Poetry	Homer's *Iliad* (8th century BCE)
Lyric Poetry	Pindar's *Olympian Odes* (5th century BCE)
Drama	Sophocles' *Oedipus Rex* (5th century BCE)
Mythology	Apollodorus' *Library* (2nd century BCE)
Literature/Parody	Lucian's *True History* (2nd century CE)
Religious Texts	The *Homeric Hymns* (7th–6th century BCE) and *New Testament* (1st century CE)
Scientific Works	Euclid's *Elements* (3rd century BCE)
Geography	Strabo's *Geography* (1st century BCE–1st century CE)

Table 4.2: Ancient Greek texts included in the dataset.

morphology through the use of ancient Greek roots and classical inflections (Horrocks, 2014).

Although a detailed exploration of the '*language question*' lies outside the scope of this study, it is crucial to note that Katharevousa was born as a response to this issue, which revolved around whether the spoken SMG language (or Demotic) could serve as the foundation for written language in laws, governance, and education in an independent Greek state (Horrocks, 2014)[5].

Katharevousa was an artificial language [6], intended as an intermediary between AG and Demotic (SMG, which was the spoken language). As such, it combined elements from both. Its foundational principle, championed by its '*father*' Adamantios Korais (1748–1833), was the synthesis of national self-determination, anchored in a connection to AG and the alignment of written language with contemporary spoken forms. Korais proposed a pragmatic program of '*correcting*' the spoken language by eliminating features that deviated from its ancient roots.

This artificial nature of Katharevousa is reflected in its goals: restoring orthography, replacing loanwords, enriching the lexicon, and reintroducing ancient words either directly or via print sources. However, it is essential to note that no native speakers of Katharevousa existed, as it was never used as a spoken language.

From a linguistic perspective, the '*corrections*' introduced in Katharevousa, particularly those proposed by Korais, primarily involved the reintroduction of ancient lexical units and the restoration of ancient orthography and morphosyntax in words and structures still in use at the time. Its nominal and verbal morphology was heavily inspired by ancient forms. For instance, compound verbs exhibited external augment (e.g., '*ἐπαρηγόρησε', 'consoled'*), a feature absent

[5]The intensity of the debates around this issue was so significant that it became the subject of satire in contemporary literature, as illustrated in Dimitrios Byzantios' play Babylonia (Βυζάντιος, 1876).

[6]It was constructed rather than naturally evolved from the speech of native speakers.

in spoken Greek. Additionally, formal prepositions replaced colloquial equivalents and were used with ancient case structures e.g., προ+ γενική *(pro + genitive)*, except in cases where such replacements were artificial revivals of Ancient Greek e.g., εις+ αιτιατική *(eis + accusative)* instead of εν + δοτική *(en + dative)*, and με + αιτιατική *(me + accusative)* was retained over the dative (Horrocks, 2014).

The following excerpt from Korais illustrates these '*corrections*'. Except for participial forms (e.g., γεννήσας, σπουδάσας), most of the syntactic structure corresponds to cultivated spoken Greek of his time, while deviations are simple substitutions (e.g., '*dioti*' instead of '*giati*', '*ostis*' instead of '*pou*', '*pro*' instead of '*prin apo*', '*eti*' instead of '*akoma*'). Korais avoided archaic forms like infinitives and datives, which had disappeared from the spoken language, and restricted corrections to the reintroduction of ancient lexical items (Horrocks, 2014). Browning (1983) notes the artificial nature of Katharevousa, emphasizing that Korais merely adjusted words like '*psari*' (fish) to '*opsarion*' (fish), highlighting the lexical replacements that defined the Katharevousa framework.

Excerpt from Κοραής (1964): Ἡ μήτηρ μου ἔλαβεν ἐλευθερωτέραν ἀνατροφήν, διότι εὐτύχησε νὰ ἔχῃ πατέρα Ἀδαμάντιον τὸν Ρύσιον, τὸν σοφώτατον ἐκείνου τοῦ καιροῦ εἰς τὴν ἑλληνικὴν φιλολογίαν ἄνδρα, ὅστις ἀπέθανεν ἓν ἔτος (1747) πρὸ τῆς γεννήσεώς μου. Αὐτὸς ἐχρημάτισεν, ἔτι νέος ὤν, διδάσκαλος τῆς ἑλληνικῆς φιλολογίας εἰς Χίον· μετὰ ταῦτα ἦλθεν εἰς Σμύρνην, ὅπου ἐνυμφεύθη χήραν τινὰ Αγκυρανήν. Οὗτος μὴ γεννήσας ἀρσενικόν, ἐπαρηγόρησε τὴν ἀποτυχίαν του, σπουδάσας νὰ ἀναθρέψῃ ὡς υἱοὺς τὰς τέσσαρας θυγατέρας του.΄

Based on the above, the features of Katharevousa that differentiate it from SMG include the polytonic system, orthography, morphological endings, individual archaic words and morphemes that replace lexical units and morphemes of Demotic Greek. In contrast, the features of Katharevousa that distinguish it from AG are syntactic structures, closer to SMG.

The confrontation over the '*language question*', which has clear ideological underpinnings, was intense, and Katharevousa sparked reactions from both supporters of Demotic Greek and proponents of Ancient Greek.

4.4 The dataset

Creating datasets for computational study is essential for conducting rigorous linguistic analysis and enabling robust data-driven research (Abney & Bird, 2010; Bird et al., 2009). In this context, a corpus refers to the collection of real-world text used for the study of natural language, primarily to investigate its statistical properties and underlying patterns.

The data collected for this study comprises real text from two sources for Katharevousa, one source for Ancient Greek, and open-access data for Standard Modern Greek obtained from the GRDD project [7]

The Katharevousa corpus was sourced from Project Gutenberg [8] and the Myriobiblos digital library [9]. For the ancient Greek language, texts from the Perseus collection [10] which includes Ancient Greek texts and those from the Roman period, was downloaded using a *python* script for web scraping.

Determining which texts to use for Ancient Greek was a straightforward process due to the comprehensive nature of the Perseus source. However, identifying representative texts for Katharevousa required a more nuanced approach. All relevant texts in Katharevousa were collected manually, including translated literary works from foreign originals, historical documents, prose literature, and religious writings. The classification of these texts as '*texts written in Katharevousa*' was based on their linguistic characteristics (Horrocks, 2014) [11].

The Katharevousa dataset is considered representative only under the assumption that the selected texts align with the linguistic definition of Katharevousa. The majority of these texts were obtained from Project Gutenberg [12]. Among the texts categorized as Katharevousa's texts, and based on the discussion in the previous section which outlines its characteristics, a selection of texts was compiled ranging from the early 19th century to 1976. Specifically, the following texts, among others, are included: Evangelidis (1894), Karkavitsas (1896), Marmontel (1845),

[7]For Standard Modern Greek, the dataset is available at https://github.com/StergiosCha/Greek_dialect_corpus/tree/main/SMG, and the processed version from the GRDD project Chatzikyriakidis et al., 2023 was used.

[8]https://www.gutenberg.org/

[9]https://www.myriobiblos.gr/library/home.htm

[10]This database currently contains 13,507,448 words from Greek texts and is an open-source project that provides a range of services for interacting with text collections: https://www.perseus.tufts.edu/hopper/collection?collection=Perseus:corpus:perseus,Greek%20Texts. The texts used in this study, however, were collected specifically from the repository available at https://github.com/PerseusDL/canonical-greekLit/tree/master (accessed on 06/11/2024).

[11]These texts remain available for review and are open to different interpretations.

[12]Texts used during liturgical practices and writings by Church Fathers, such as John Chrysostom and Basil the Great, were excluded as they reflect Medieval Greek characteristics. Regarding the Myriobiblos texts, many were ultimately excluded because they contained Ancient, Medieval, or Modern Greek rather than Katharevousa. Exceptions were works by authors like Vikelas and Nirvanas, which met the criteria, but were limited in number

Shakespeare (1889), Trikoupis (1860-1862), Vlachos (1901), and Βιζυηνός (1883).

Three distinct datasets were created: one for Ancient Greek, one for Katharevousa, and one for Standard Modern Greek. Pre-processing and cleaning procedures were applied to all texts to ensure uniformity and facilitate computational analysis. These procedures, implemented using *python* and command-line tools, included the removal of punctuation, white spaces, special symbols, and special characters. Empty lines were removed, and line lengths were standardized to 56-73 characters. Additionally, all texts were transformed from a polytonic to a monotonic system, and all characters were converted to lowercase. Duplicates were also removed to avoid redundancy in the dataset.

The word counts [13] for each cleaned dataset are presented in Table 4.3.

Text Category	**Word Count**
Ancient Greek	3,997,550
Katharevousa	1,516,327
Standard Modern Greek	4,532,026

Table 4.3: Word count for each cleaned dataset: Ancient Greek, Katharevousa, and Standard Modern Greek.

4.5 Basic Concepts: Machine Learning

Before discussing the implementation of the Language Identification task in this study, it is essential to define some basic concepts and describe how our methods operate. To begin with, what do we mean by the term *Machine Learning* (ML)? Machine Learning is defined as a subset of Artificial Intelligence (AI) that enables computers to *learn* from data and make decisions or predictions without being explicitly programmed for that task. In ML, systems use data to *train* themselves to recognize patterns, thereby enabling them to predict future outcomes. Machine Learning algorithms can be applied in a wide range of applications, including image recognition, text analysis, trend prediction, and language identification (Jurafsky & Martin, 2024; Müller & Guido, 2016).

One of the primary tasks of Machine Learning, often referred to as an *umbrella task* due to the variety of problems it encompasses, is classification. Classification is at the core of both

[13] Although we are aware of the imbalance between words in each category, the large number of words allows the algorithms to be trained to a minimum of 1.5 million of words, which, as we know from other cases of Greek dataset creation, may be sufficient for the algorithm to distinguish between categories (Chatzikyriakidis et al., 2023). In general, the greater the number of words, the more effective the results, which is why we did not conduct tests on smaller datasets, but instead utilized the entire range of words available to us.

human cognition and computational reasoning. It involves determining when an input x belongs to a broader category or, in other words, assigning x to a specific class. Generally speaking, the goal of classification is to predict a label of *class* selected from a predefined list of possible choices. This classification can be either binary or multiclass (as in our case, where we have a choice among three languages) (Jurafsky & Martin, 2024). Within this concept, we encounter a variety of tasks, such as text categorization, sentiment analysis, spam detection, and authorship attribution.

A particular form of classification is text classification, which aims to categorize text into meaningful categories such as topics, sentiments, or language varieties. Within this framework, Language Identification emerges as a subtask of text classification, where the aim is to determine the language of a given piece of text (for more on the differentiation between text classification and language identification, see Jauhiainen et al. (2019)).

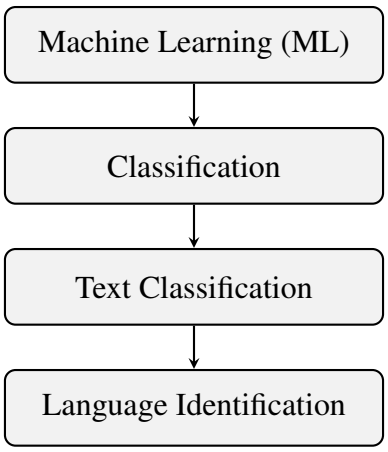


Figure 4.1: Hierarchy of tasks: Machine Learning → Classification → Text Classification → Language Identification.

4.5.1 Classification and Language Identification

The primary aim of classification is to extract meaningful features from an observation and categorize it into one of a set of distinct classes. A particularly common way to achieve classification is through *supervised machine learning*. In supervised learning, we begin with a set of input observations (e.g. sentences in SMG, Katharevousa, or AG), each associated with a correct output (e.g., the language labels: '*smg*,' '*kat*,' or ' *ag*'). This process provides a ' *supervised* ' signal, and the algorithm learns to map unseen observations to their correct outputs. Simply put, the algorithm learns to identify patterns and uses this knowledge to classify new,

unseen data (Jurafsky & Martin, 2024; Müller & Guido, 2016). Building on this, Language identification is a specific application of classification in which the goal is to determine the language of a given piece of natural language data (Jauhiainen et al., 2019). In our case, the task is successful if unseen text is correctly assigned to one of the three predefined language categories (SMG, KAT, or AG), as determined by the model trained on labeled examples.

4.5.2 N-grams and Probabilistic Models

In order to explain the algorithm that performs the Language Identification task, it is useful to discuss *n-grams*, which play a key role in the extraction of feature. The foundational mathematics of n-gram models, initially proposed by Markov in 1913 as mentioned in Jurafsky and Martin, 2024, heavily involve probability theory. To avoid any term ambiguity, n-grams generally refer to either probabilistic models estimating the likelihood of a word based on the n-previous words, thus assigning probabilities to an entire sentence, or to sequences of n consecutive words. In this discussion, we focus on n-grams as sequences of n words, as this concept is crucial in training text classification algorithms.

Now, what do we mean by a sequence of words? The consecutive word sequences in a text include examples such as '*Alexandros Papadiamantis*' (2-gram), '*Papadiamantis wrote stories*' (3-gram), or '*Papadiamantis wrote beautiful stories*' (4-gram). These groupings of words directly relate to probabilistic n-gram models, as they define the frequency of word sequences and contribute to training classification algorithms.

In order to understand the importance of n-grams as sequences of words, Jurafsky and Martin (2024) notes that n-gram language models are trained to predict the next word by assigning probabilities based on preceding words. For instance, a 2-gram model ($P(w_n \mid w_{n-1})$, where w_{n-1} is the preceding word and w_n the predicted word estimates the likelihood of the word *stories* (w_n) following *wrote* (w_{n-1}) based on observed frequencies in the dataset. Probability estimation is performed using Maximum Likelihood Estimation (MLE), where the counts are normalized to a range between 0 and 1.

For example, if we wanted to calculate the probability of the word *stories* (w_n) appearing after *wrote* (w_{n-1}), we would count the occurrences of the bigram *wrote stories* ($C(w_{n-1}w_n)$) and divide this by the total occurrences of *wrote* followed by any word ($\sum_w C(w_{n-1}w)$). This result is then normalized to a value between 0 and 1.

$$P(w_n \mid w_{n-1}) = \frac{C(w_{n-1}w_n)}{\sum_w C(w_{n-1}w)}$$

Even though n-grams are a simple concept, they are essential for the current study, as they are used to extract features for each language and to provide input during the algorithm's training procedure.

4.5.3 Naive Bayes Classifier

As previously mentioned, the approach used to tackle the Language Identification task in this study is based on classic ML algorithms, specifically supervised ML algorithms. In supervised learning, we have a dataset consisting of input observations, each associated with the correct output. Here, the labels correspond to the language categories: Standard Modern Greek (smg), Ancient Greek (ag), or Katharevousa (kat). The algorithm's goal is to learn how to map a new, unseen observation to the correct output (Jurafsky & Martin, 2024).

In this form of supervised learning, the focus lies on constructing a model based on training data, which can then make accurate predictions on new unseen data with characteristics similar to those of the training set (Müller & Guido, 2016). If the model can make accurate predictions, it demonstrates its ability to generalize knowledge from the training set to the test set.

There are two types of classifiers commonly used: *Generative classifiers*, like Naive Bayes, and *Discriminative classifiers*, like logistic regression. Generative classifiers build a model of how a class can generate some input data. Given an observation, they return the class most likely to generate that observation. In contrast, discriminative classifiers focus on identifying which features of the input are most useful for distinguishing between possible classes (Jurafsky & Martin, 2024).

For the scope of this work, it was decided to emphasize the function and evaluation of the algorithm that performed best on the required task. Specifically, the algorithm with the best performance in this study was the *Multinomial Naive Bayes Classifier*, a type of Bayesian classifier. The Naive Bayes classifier makes a '*naive*' assumption about how features interact. Naive Bayes algorithms provide a lightweight and efficient solution that performs competitively in many text classification scenarios, particularly when computational simplicity is a priority. The concept of Bayesian inference, as mentioned in Jurafsky and Martin (2024), originates from Bayes (1763), and was first applied in text classification tasks by Mosteller and Wallace (1964).

Two key principles distinguish this type of algorithm: the Bag of Words assumption and the Naive Bayes assumption.

Under the Bag of Words (BoW) assumption, Naive Bayes classifiers treat a text as an unordered set of words, ignoring the order of the words and considering only their frequency within the document. For example, consider the following text:

> '*I arrived at the university today to take my final exam for graduation. Without much thought, I rushed in the morning and took the first bus to the university. The professor distributed the exam topics. My success was certain; the degree was close, and I felt that my joy illuminated the entire lecture hall.*'

The Bag of Words assumption would treat the text as follows (ignoring punctuation and order of words):

- '*the*': 7 times
- '*my*': 3 times
- '*was*': 2 times
- '*degree*': 1 time
- ...

This approach emphasizes the frequency of words without considering their sequential arrangement.

Moving on to the next key point of Naive Bayes, one might wonder why the algorithm is characterized as '*naive*'. Naive Bayes is called '*naive*' because it makes a simplifying assumption: it assumes that the features f_i are conditionally independent given the class c. A class refers to the category to which an input belongs. For example, an email might be classified as spam or non-spam, or a sentence might be categorised as Greek, Cypriot, or Pontic. This allows the probabilities $P(f_1, f_2, ..., f_n|c)$ to be calculated as:

$$P(f_1, f_2, ..., f_n|c) = P(f_1|c) \cdot P(f_2|c) \cdot ... \cdot P(f_n|c)$$

Despite this '*naive*' assumption, Naive Bayes classifiers often perform remarkably well, particularly for text classification tasks. By combining probabilistic reasoning with the independence assumption, Naive Bayes provides an efficient and interpretable model for solving

the Language Identification task. Therefore, these two characteristics ultimately constitute a fundamental aspect of the algorithm's functionality (and success), namely that the algorithm avoids the unnecessary complexity of calculating a multitude of parameters and large training sets, which would take into account both the position of a word in a text and all possible combinations of words.

The Naive Bayes classifier is a probabilistic classifier, meaning it predicts the class $\hat{c}$ for a given document d based on the highest posterior probability among all classes c in C. Mathematically, we can see the following equation:

$$\hat{c} = \arg\max_{c \in C} P(c \mid d)$$

Here, $\arg\max$ refers to selecting the argument c that maximizes the function $P(c \mid d)$. This function represents the probability of class c given the document d. By Bayes' theorem, we can break down $P(c \mid d)$ into three probabilities, by combining the prior probabilities $P(c)$ with the likelihoods $P(d \mid c)$:

$$P(c|d) = \frac{P(d|c) \cdot P(c)}{P(d)}$$

In fact, the above theorem tells us that if we want to calculate the probability of a class *c* occurring, given a text, it will equal the ratio of the product of the probability of the text (its features) occurring given the class, multiplied by the probability of the class occurring, divided by the probability of the text occurring

Applying this reasoning to a Language Identification task, let us consider a scenario where we want to compute the probability of a text x belonging to the Katharevousa category by using Bayes' theorem:

$$P(\text{Katharevousa}|x) = \frac{P(x|\text{Katharevousa}) \cdot P(\text{Katharevousa})}{P(x)}$$

The important point here is that this calculation incorporates both the observations made by the model ($P(x|\text{Katharevousa})$) and the class distribution in the dataset ($P(\text{Katharevousa})$). This approach allows the prior probabilities to be effectively combined with the posterior probabilities.

Consider a concrete example from the dataset. Suppose we want to determine the probability

that the text x, ' Υμνώ μετ έρωτος την φύσιν ' belongs to the category Katharevousa[14]. We can explain explain each probability separately. First, the probability $P(x|\text{Katharevousa})$ measures how often these words appear in the training data for Katharevousa. To compute this, we multiply the individual probabilities of each word in the text and divide the result by the total frequency of all features. Next, the prior probability $P(\text{Katharevousa})$, representing the likelihood of encountering the Katharevousa category in the dataset, is calculated as:

$$P(\text{Katharevousa}) = \frac{\text{Number of Katharevousa texts}}{\text{Total number of texts}}$$

Finally, $P(x)$, the total probability of observing text x, is computed by summing over all possible categories:

$$\begin{aligned} P(x) = {} & P(x|\text{SMG}) \cdot P(\text{SMG}) \\ & + P(x|\text{Katharevousa}) \cdot P(\text{Katharevousa}) \\ & + P(x|\text{Ancient Greek}) \cdot P(\text{Ancient Greek}) \end{aligned}$$

This process allows us to calculate $P(\text{Katharevousa}|x)$, the probability of x belonging to the Katharevousa category. To conclude, if we substitute the above probabilities, we could calculate, using the terms of the Naive Bayes theorem, the probability of any phrase belonging to each category.

In addition, the concept of decision-making is very important for the algorithm. In the Naive Bayes classification approach, converting probabilities to logarithms is a strategic decision that significantly enhances computational efficiency. This transformation is essential, especially when dealing with small probabilities that can lead to computational issues if processed directly. The classification decision is made by first calculating the sum of logarithmic values of prior probabilities ($\log P(c)$) and the logarithmic likelihoods of observing each feature given the class ($\log P(w_i|c)$), for all features i. Then, the class that accumulates the highest total from these logarithmic values is selected as the most likely class for the input data (Jurafsky & Martin, 2024).

This method effectively changes the computational model from a multiplicative framework

[14]As text features, we assume here that the word frequency in the dataset is considered, rather than 2-grams or 3-grams.

to an additive one. By doing so, it simplifies the calculations and minimizes the risk of numerical errors, such as underflow, that can occur with very small numbers. Additionally, this linear transformation in the log space ensures that the classifier can efficiently and accurately process large datasets, making it particularly advantageous for applications like text processing. The adoption of a linear additive model in the decision-making stage not only boosts computational speed but also enhances the overall reliability and precision of the classification process (Jurafsky & Martin, 2024).

Another one crucial issue in the operation of the Naive Bayes algorithm is managing unknown words and the zero probabilities they introduce into probability calculations. This problem arises when words do not appear in the algorithm's training set. Additionally, some words might appear in the training set only within one category, resulting in a zero probability for all other categories, even though they may occur in other categories in the test set.

To address unknown words, a common approach is to ignore them during the probability calculations. This prevents their zero probability from nullifying the product of probabilities for a given class. For words that appear only in one category in the training set, a method called smoothing, such as Laplace [15] smoothing, is employed. This technique adds one to the count of each word, adjusting the denominator to account for the increase in total observations. The probability is then calculated as:

$$P_{\text{Laplace}}(w_i) = \frac{c_i + 1}{N + V}$$

where c_i is the count of the word w_i, N is the total number of word occurrences in the dataset, and V is the size of the vocabulary. This approach ensures that no probability is exactly zero, enabling more robust classification results (Jurafsky & Martin, 2024).

[15]Laplace Smoothing, also known as add-one smoothing, is a critical technique in natural language processing and text categorization, particularly for addressing cases where zero counts occur. This method increases the count of each n-gram by one before these are converted into probabilities, thereby transforming zero values to ones and incrementing existing counts by one. When calculating probabilities, adding the total number of words in the vocabulary to the denominator prevents the additional units from disproportionately influencing the probabilities. Laplace Smoothing is especially useful for providing a comparative baseline among more complex smoothing algorithms and for understanding the fundamental principles that govern smoothing in language models, thus offering a practical way to enhance performance in text categorization (Jurafsky & Martin, 2024).

4.6 Tools and Libraries for Training Machine Learning Algorithms

Let us now consider the training of ML algorithms and the steps that we followed in order to train them. ML algorithms were trained and evaluated using the Jupyter Notebook [16] environment within the Anaconda platform and implemented in *Python* [17]. To perform the training, the following libraries were utilized (Müller & Guido, 2016):

- *Pandas*: This library was used for data analysis and manipulation. It facilitated reading text files, storing data in dataframes, and merging datasets into a single dataframe for further processing.
- *Scikit-learn*: The core library for implementing machine learning algorithms, feature extraction, splitting datasets into training and testing subsets, and evaluating algorithm performance.
- *Seaborn* and *Matplotlib*: Used for data visualization, these libraries helped create graphical representations, including heatmaps for confusion matrices and other performance metrics.
- *Time*: Utilized to measure the time required for training and testing each algorithm.

4.6.1 Steps in Training and Feature Extraction

After importing the necessary libraries, the categories *smg*, *kat*, and *ag* were defined. Text files containing labeled datasets were loaded, where each line included a text segment followed by a delimiter (;) and the language label (*gold label*) indicating the actual category of each text. The datasets were concatenated into a single dataframe and then split into training (80%) and testing (20%) subsets using the *train_test_split* method from scikit-learn.

In addition, feature extraction was conducted using the TF-IDF Vectorizer (Term Frequency-Inverse Document Frequency), which transformed the text data into numerical features. This step allowed the algorithm to process text efficiently. The vectorizer was configured with the following parameters:

- `analyzer='word'`: Analyzed text at the word level.

[16] https://anaconda.org/anaconda/jupyter
[17] https://www.python.org

- `ngram_range=(2,3)`: Extracted 2-grams and 3-grams as features.
- `max_df=0.5`: Ignored terms appearing in more than 50% of the documents to filter out stopwords.

The `fit_transform` method trained the vectorizer on the training data (`X_train`) and created a sparse matrix where each row corresponded to a document and each column to a feature. For the test data (`X_test`), the same vectorizer was used to ensure consistency. The output statistics included the number of samples and features extracted:

```
Extracting features from the training data using a sparse vectorizer
done in 116.047960s at 0.756MB/s
n_samples: 678555, n_features: 8313951

Extracting features from the test data using the same vectorizer
done in 5.284926s at 4.148MB/s
n_samples: 169639, n_features: 8313951
```

The standard process followed both during the training phase (a) and during the prediction phase (b) is presented in the image below:

Specifically, during training, a feature extractor is used to convert each input value into a feature set. These exact feature sets capture the basic information about each input that should be used to classify it. Pairs of feature sets and labels are fed into the ML algorithm to generate a model. On the other hand, during prediction, the same feature extractor is used to convert unseen inputs into feature sets. These feature sets are then fed into the model, which generates predicted labels (Müller & Guido, 2016).

In the training phase, several classifiers were benchmarked, including Ridge Classifier, Perceptron, Passive Aggressive Classifier, Naive Bayes (MultinomialNB and BernoulliNB), SGDClassifier, Nearest Centroid, and LinearSVC.

The training involved fitting each algorithm to the training data (`X_train`, `y_train`) and evaluating their predictions on the test data (`X_test`). Subsequently, metrics such as *accuracy score*, *classification reports*, and *confusion matrices* were computed to assess performance. Also, visualization tools, like `ConfusionMatrixDisplay` and `heatmap`, were used to represent results graphically. Finally, training and prediction times were measured for each classifier.

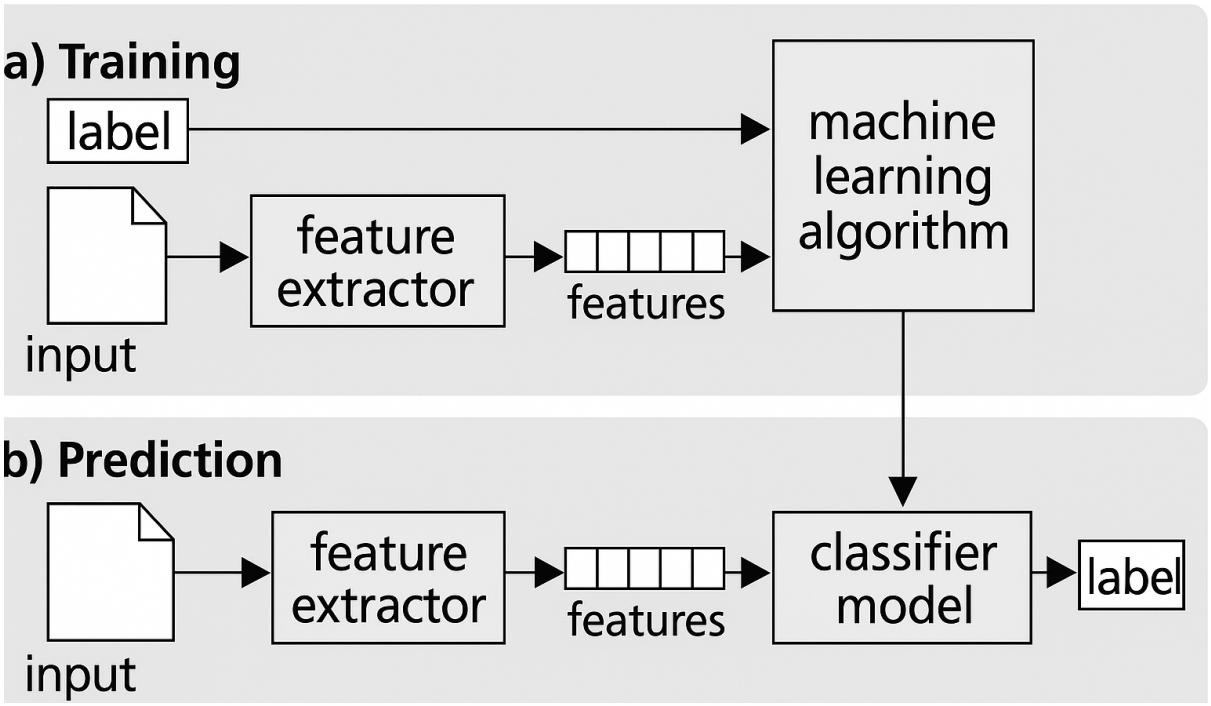


Figure 4.2: This diagram is adapted from Müller and Guido, 2016, which details the methodology used in the training and prediction process.

In general, the results of these steps helped identify the best performing algorithm of the Language Identification task. This training process ensured that the chosen classifier, provided high accuracy and efficiency in classifying text into *smg*, *kat*, and *ag* categories. So far, we have seen the training process of the algorithms and how it is carried out. Starting from the text, features are extracted that are in a suitable form for the algorithm to process, and these features are used by the algorithm during its training. Having trained a number of algorithms, from this point forward, we will focus on the one that performed best on this task, the *Multinomial Naive Bayes (Alpha=0.01)*. The next section presents the results and further analysis of the classifiers' ability to correctly classify the given datasets.

4.7 Results

The table bellow 4.4 lists the top 3 classifiers based on the F1-Score (it will be explained below why this particular measurement is crucial in our choice) for the task of language identification between SMG, AG, and Katharevousa.

These metrics demonstrate the effectiveness of the Naive Bayes algorithms, with Multinomi-

Table 4.4: Performance Metrics of the Top 3 Algorithms for Language Identification

Algorithm	Macro Avg. F1-Score	Weighted Avg. F1-Score	Accuracy
MultinomialNB (Alpha=0.01)	**0.96**	**0.97**	**0.971**
BernoulliNB (Alpha=0.01)	0.96	0.97	0.970
MultinomialNB (Alpha=0.1)	0.96	0.97	0.968

alNB (alpha = 0.01) achieving the highest accuracy among the variants [18] in recognizing distinct linguistic features accurately, ensuring high precision and recall across different types of Greek language texts. As shown in the table 4.4, this particular algorithm had the best performance, and for this reason, we are focusing on its evaluation and performance.

To set the stage for this section, we will discuss the performance results of the best-performing algorithm, *Multinomial Naive Bayes (Alpha=0.01)*. To do so, it is essential to understand the metrics required to evaluate an algorithm's performance and determine how the classifier fares across these metrics.

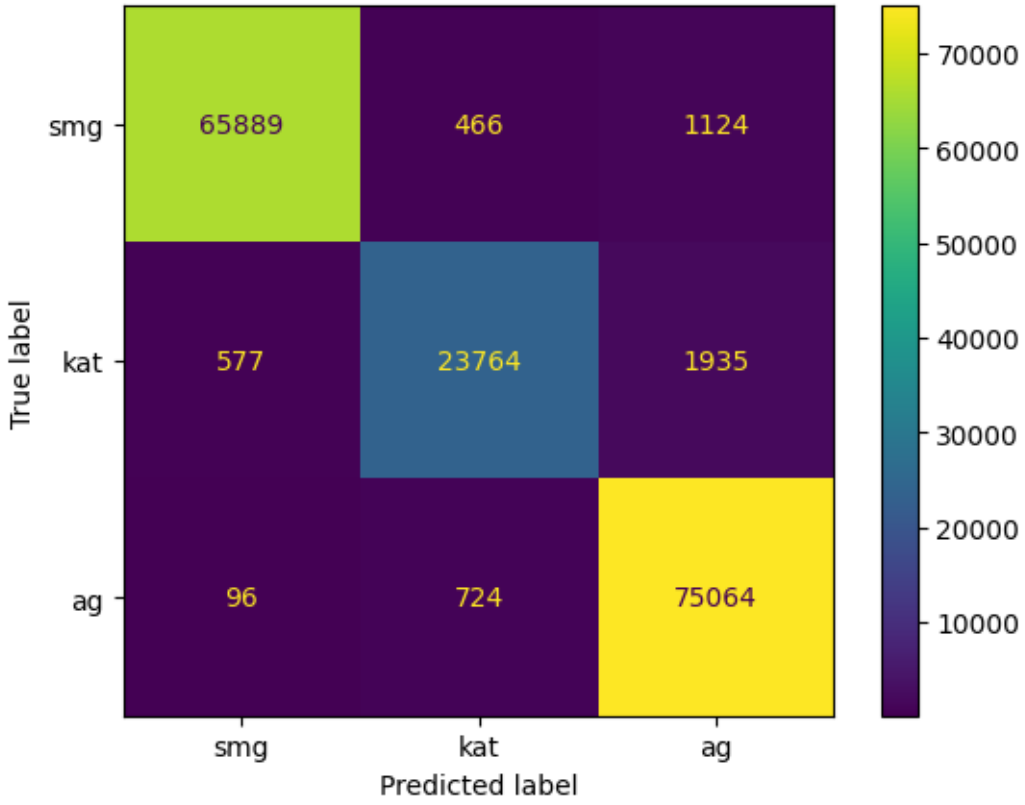
When evaluating a system like the language detector among the three previously mentioned languages, we start by building a confusion matrix. This is a table that represents how the algorithm performs relative to the gold labels, using two dimensions: the system's output and the gold labels. Each cell in the matrix corresponds to a specific pairing of predicted and actual labels. Thus, in the confusion matrix of MultinomialNB (Alpha=0.01), we see how the actual labels of each language correspond to the cases, with the labels that the algorithm predicted these

Parameter	Value
MultinomialNB (alpha)	0.01
Training Time	2.126s
Testing Time	0.093s
Accuracy	0.971

[18] The two versions of the Multinomial Naive Bayes differ only in the value of the alpha parameter. The alpha factor is used to smooth the model, reducing the impact of features (such as words) that may not appear at all in some training samples. The first classifier, which performs best, uses an alpha value of 0.01. This small alpha value enhances the smoothing effect, reducing the likelihood that rare features (words with low frequency) have a negative impact on the model's accuracy. An alpha of 0.01 allows the model to be more careful with words that appear less frequently. The Multinomial Naive Bayes algorithm is a variant of Naive Bayes that is primarily used in problems where features are frequencies or counts, such as in text classification problems. In this case, the features represent how often each bigram-trigram appears in a document. Multinomial Naive Bayes uses Bayes' theorem to calculate the probability of a category and fits well in classification problems with multiple categories and for large data sets, such as in the classification of electronic messages.

MultinomialNB and BernoulliNB utilize a singular parameter, alpha, to manage model complexity. Alpha functions by adding a number of virtual data points equivalent to its value to the dataset, where these points possess positive values across all features. This addition effectively smooths the statistical output. A higher alpha value leads to greater smoothing, thereby simplifying the model. While the performance of the algorithm is generally stable across different alpha settings, indicating that precise tuning of alpha is not essential for achieving good results, fine-tuning alpha can often enhance accuracy slightly (Müller & Guido, 2016)

cases belong to. Therefore, each row contains the actual labels, while each column contains the labels predicted by the classifier:



Starting with the observations from the above confusion matrix, for SMG, the algorithm had a high success rate, correctly predicting 65,889 cases during testing. It was most confused when evaluating cases from AG (1,124) and less so from Katharevousa (466).

Regarding Katharevousa, the classifier correctly predicted the lowest number of cases compared to the other two languages (23,764), mainly confusing them with cases from AG (1,935) and less so from SMG (577). At the same time, algorithms such as LinearSVC (dual=False, tol=0.001) (21,655) and LinearSVC(dual=False, penalty=L1, tol=0.001) (21,017) had a lower number of predictions as correct for the label of the Katharevousa, with f1-scores accuracy of 0.95 and 0.93 respectively, indicating a challenge that the algorithms had to overcome, namely to distinguish a language variety that has elements from two other varieties. Based also on this data, MultinomialNB (Alpha=0.01) seems to perform much better in classifying Katharevousa as Katharevousa.

Finally, for AG, the algorithm had the best prediction performance (75,064), with relatively low confusion with Katharevousa (724) and minimal confusion with SMG (96).

Regarding the above, we can say the following about the classification of the algorithm.

Initially, concerning AG and SMG, the algorithm appears to have very good numbers of correct predictions. The lower performance in predictions for Katharevousa can be explained based on what has been said about the idiosyncrasies of this language in the relevant subsection 4.3.3. Because Katharevousa shares features with both SMG and AG, the algorithm's predictions for this variety frequently exhibit overlap and confusion. However, the truth is that, structurally, Katharevousa is SMG and borrows superficial elements from AG, such as suffixes, vocabulary, prepositions, and standard expressions. Therefore, the appearance of this image in the confusion matrix of Katharevousa as less distinct than the other two languages is most likely due to the way the Naive Bayes algorithm functions, specifically, its inability to capture deeper syntactic or more complex structural patterns. This does not imply that Katharevousa '*balances*' between the other two languages.

Continuing the evaluation of the classifier's performance, the next metrics that serve in assessing the algorithm are *accuracy*, *precision*, *recall*, and *F-measure* (Van Rijsbergen, 1975), which are included in the classification report for the classifier we are analyzing in table 4.5 below:

Class	Precision	Recall	F1-Score	Support
Greek	0.99	0.98	0.98	67479
Katharevousa	0.95	0.90	0.93	26276
Ancient_Greek	0.96	0.99	0.97	75884
Overall Performance				
Accuracy		0.971		
Macro Avg	0.97	0.96	0.96	169639
Weighted Avg	0.97	0.97	0.97	169639

Table 4.5: Classification report for the Naive Bayes classifier.

What does each of the above metrics refer to, and which are crucial for evaluating the performance of the algorithm? The *accuracy* (0.971) is the metric that shows the percentage at which the system evaluates with the correct labels (Jurafsky & Martin, 2024). Although it seems, of course, a very natural characteristic to base our interpretation on, it does not serve this role. The reason is that the algorithm has classes with an imbalance in the number of data. In such classifications, *accuracy* can operate in a way that is not representative of performance. Let's consider an example. Imagine a digital library classification task where the goal is to identify books related to the Greek author Alexandros Papadiamantis among a total of 10,000 books. Suppose only 100 of these books are about Papadiamantis, while the remaining 9,900 are on

unrelated topics. A naive classifier that categorizes every book as ' *unrelated to Papadiamantis* ' would achieve a superficial accuracy of 99% by correctly identifying the 9,900 unrelated books. However, this approach completely fails to identify any of the relevant Papadiamantis books, missing all 100 true positives. Such a scenario highlights the inadequacy of using *accuracy* as a metric in datasets, where one class is significantly underrepresented. Consequently, *accuracy* is not particularly indicative for datasets that do not have a balance of the input observations, such as ours.

Moving forward, to the basic metrics, let's see how *precision*, *recall*, and *F1* are calculated. *Precision* measures the percentage of items that the system identified (for example, categorized as Katharevousa) and that actually belonged to the category they were categorized in (truly were Katharevousa) (Jurafsky & Martin, 2024). Thus, in the current confusion matrix, if we wanted to calculate the precision for Katharevousa, we would divide 23,764 by the result of adding all categorizations as Katharevousa (724 + 23,764 + 466).

On the other hand, *recall* measures the percentage of items that were actually present in the input and were correctly recognized by the system as belonging to their true category (Jurafsky & Martin, 2024). For example, if we wanted to calculate the *recall* now in the confusion matrix for Katharevousa, we would divide the correct predictions, 23,764, by all the items that were introduced into the input as Katharevousa (577 + 23,764 + 1,935).

The *F-measure* provides a method to synthesize *precision* and *recall* into a single metric, defined as follows:

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R} \tag{4.1}$$

Here, β adjusts the relative importance of *recall* versus *precision*, tailored to specific requirements of the analysis. With $\beta = 1$, the formula balances *precision* and *recall*, referred to as the F1 score:

$$F1 = \frac{2PR}{P + R} \tag{4.2}$$

The *F-measure* is derived from the weighted harmonic mean of *precision* and *recall*. The principle behind using a harmonic mean, which is the reciprocal of the arithmetic mean of the reciprocals of the numbers, is that it tends to be closer to the smaller of the numbers being averaged. This characteristic gives greater weight to the lower value, providing a more conservative measure in assessments where both *precision* and *recall* are important but may have different weights (Jurafsky & Martin, 2024).

Therefore, the metric that will be considered for the evaluation of the algorithm is the *F1-score*, which represents the harmonic mean of *precision* and *recall*. Accordingly, as shown in the table, the classifier demonstrates a 0.98 precision in predicting SMG, 0.93 for Katharevousa, and 0.97 for AG. Both for AG and SMG, the percentages are quite close to perfect. However, for Katharevousa, we observe a decline in the *F1-score*, which at first glance can be interpreted as a result of its intermediate nature between SMG and AG, as discussed. Nonetheless, the above values are indicative that the classifier successfully achieves high accuracy in recognizing when an input item corresponds to each category. In other words, the above language categories appear to have distinctive features that the algorithm is capable of identifying and distinguishing, especially considering that it does not seem to respond randomly (which would mean a percentage close to 0.3), but with high *F1-scores*.

4.7.1 Error Analysis during Classification by the Multinomial Naive Bayes (Alpha=0.01)

A significant aspect is the error analysis of the classification performed by the algorithm, aimed at identifying the main causes related to the cases of misclassification. In order to pinpoint such instances, we have selectively identified examples where the true label does not match the predicted label for each category.

For SMG, there were cases classified as Katharevousa, which align with our common intuition, as although they were found in the SMG dataset, they seem to linguistically fit Katharevousa (archaic words, syntactic patterns that survive in written speech aimed at archaic usage of the language, texts that survive in Katharevousa mixed with Modern Greek). This can be attributed to the fact that in texts of SMG, it is not entirely unusual to find such sentences as the following:

True Label	Predicted Label	Text
smg	kat	επιθυμούσα να φύγει έτσι έχοντας ελάχιστα γευτεί τη γλύκα της ζωής και
smg	kat	ότι η σημειωθείσα αργοπορία του διά τον κίτρον και βενετίαν ιουστίνος απολ α επίστευεν ότι πάντες οι άνθρωποι εξελεύσονται των τάφων

Table 4.6: Misclassification examples in Modern Greek categorized as Katharevousa

Similarly, there were cases akin to the above, which were categorized as AG, explainable by

the presence of archaic words and usage of syntactic patterns, as previously:

True Label	Predicted Label	Text
smg	ag	πατέρων καινοτομούμενον και της πάντων ελευθερίας απ- τόμενον προσήγγειλεν
smg	ag	οι τονικότητες της ντο μείζονας και λα ελάσσονας δεν έχουν διέσεις ή

Table 4.7: Misclassification examples in Modern Greek categorized as Ancient Greek

Continuing, for Katharevousa, where the highest percentage of misclassification concerned SMG, we observe cases that could belong to SMG, with a likely explanation being their apparent similarity, despite the conscious corrections attempted in this linguistic category, discussed in the corresponding chapter 4:

True Label	Predicted Label	Text
kat	smg	ύστερα από τους περίφημους πολέμους της αράχωβας του διστόμου και τόσους
kat	smg	ευρώπη οι θρησκευτικοί πόλεμοι ή οι ληστρικοί των δι- αφόρων ηγεμόνων που η

Table 4.8: Misclassification examples in Katharevousa categorized as Modern Greek

Additionally, confusion with AG occurred for the same reason we identify common patterns with them, and the algorithm correctly got confused:

True Label	Predicted Label	Text
kat	ag	τάφρον τους εφόνευσαν οι δε προλαβόντες και διαβάντες οι και πλειότεροι
kat	ag	όταν αφίνωσι τα καθάρματα της πόλεως εις ύπαιθρον συμπόσιον των τίποτε
kat	ag	αιρεσιάρχου αρείου πρεσβυτέρου της των αλεξανδρέων εκκλησίας ο

Table 4.9: Cases where Katharevousa was confused with Ancient Greek

Generally, the confusion in this category does not seem to be random, as when the algorithm gets confused, it attributes to the category that the sentence could arguably belong to, given the close relationship of Katharevousa with the other two languages.

Finally, for AG, where the confusion occurred mainly with Katharevousa and less with SMG, the examples show that really both words and syntactic patterns (co-occurrence of words) could be contained in a Katharevousa context:

True Label	Predicted Label	Text
ag	kat	εμίσει τον σωκράτην ο κριτίας ώστε και ότε των τριάκοντα ων νομοθέτης
ag	kat	γραια έρανος σιμονίδης έλεγε προς τους εγκαλουντας αυτ ω φιλαργυρίαν ότι

Table 4.10: Cases where Ancient Greek was confused with Katharevousa

For SMG, we consider the 96 errors that occurred in this prediction category to be real mistakes, and the only explanation can be linked to individual words, which are found in both SMG and AG, but speaking safely, these occurred at a negligible rate:

True Label	Predicted Label	Text
ag	smg	θυσία και τη λειτουργία της πίστεως χαίρων και συγχαίρων τους δέ προς ους
ag	smg	μητέρα φερσεφόνης ταυροφόνων δ αμέγαρτα και αιμαλέα κρέα δόρπων θηρσι
ag	smg	τραπέζης κνίση λίπει θυλήμασιν θεων έδαισε ρινας ουδ ειπε μουνον ήλιτον

Table 4.11: Real misclassification errors in Ancient Greek categorized as Modern Greek

4.8 Predictions: Applying the Multinomial Naive Bayes Algorithm (Alpha=0.01) to Papadiamantis' Text

The next step, after saving the algorithm and the vectorizer, was to apply it to both the dialogic and narrative parts of the Papadiamantis corpus, specifically to the collection of stories and novels [19].

However, what constitutes a narrative and what constitutes a dialogic part? How are they identified and isolated? How are they grouped into separate text files? What form must they take in order to be processed by the algorithm?

In order to identify and distinguish the dialogic from the narrative parts of the Papadiamantis corpus, specifically the stories and novels, a regular expression was used with the *python*, which locates and returns all the contexts that can be considered as containing dialogue and the contexts defined as narrative.

To distinguish dialogic contexts from narrative ones, a function was used with the purpose

[19]Poetry is not examined in this specific task on the rationale that poetry as a genre does not satisfy the distinction between narrative and dialogic parts.

of compiling into a list all lines that can be considered dialogic and those that can be considered narrative. The dialogic lines were gathered based on whether they start with a dash (-), where if they satisfy this variable they are added to the dialogic parts; if not, to the narrative parts. For example, in the text below, the function would add to the dialog list the parts ('Hello! How are you? - Fine, and you?') and to the narrative parts ('This is a narrative. This is another narrative.'):

This is a narrative.

-Hello! How are you?

-Fine, and you?

This is another narrative.

A limitation of this dialog detection can be considered that it does not collect lines if they are dialogic and do not start with a dash (-) as in:

'Τι λόγον να δώση η Φατμά εις τον κύριόν της, αφού ούτος τη είχεν εμπιστευθή την νέαν εκείνην σεβιλμέκ (τήν αγαπωμένην) και τη είχεν ειπεί: 'Μού αποκρίνεσαι, κουζούμ, με το κεφάλι σου'· τι λόγον να δώση, άν ο κύριός της ήρχετο την στιγμήν ταύτην, ως και έμελλε να έλθη, νά τήν ερωτήση: - Που είναι εκείνη, Φατμά;' (Παπαδιαμάντης, Αλέξανδρος, 1885/2005)

In the above excerpt, the dash (-) is not at the beginning of the line, while the dialogic part starts after a colon and quotation marks, a pattern that would not be suitable to be defined as a general function for locating dialogic parts, as it is not as widely used as the dash at the beginning of a sentence in the Papadiamantis corpus.

It is therefore important to note that the extent of the dialogic and narrative parts in terms of words represents that portion which the editorial team recognizes as dialogic and narrative through the use of a dash (-) at the beginning of the line until the next line that indicates a semantic end [20]. If for some reason [21] dashes are not used, then the part is not characterized as dialogic, despite the fact that it may actually be so. Therefore, these observations constitute limitations of this methodology, which are taken into account in the results presented in the table below regarding the percentage of language used in the narrative and dialogic parts of Papadiamantis' work, in novels and short stories.

Thus, after applying the function, two files were created: one containing the dialogues and one containing the narrative parts from both the novels and the short stories. Following the

[20] This approach excludes any continuation of the dialogue onto lines that do not begin with a dash.

[21] Omission, or, editorial choice, or use of dialogic parts as direct speech in indirect speech contexts etc.

necessary preprocessing steps[22] to prepare the text files for application of the ML algorithm, the total size of each category is presented in the table below:

Category	**Dialogic Parts**	**Narrative Parts**	**Total Words**
Short Stories	38,104	419,875	466,977
Novels	41,510	147,091	195,720

Table 4.12: Distribution of dialogic and narrative parts in short stories and novels

In our analysis, the calculation of percentages was performed based on the total length of the entries, as delineated by the '*Length*' column in our results DataFrame. We started by determining the total sum of this column to ascertain the comprehensive magnitude of the data under review. Following this, the data was aggregated based on the '*Predicted Label*', and the sum of '*Length*' for each category was computed. This aggregation facilitated an understanding of the proportional contribution of each category within the entirety of the dataset. These sums were then converted into percentages of the total to shed light on the relative prominence of each category. So, by this way we can have a clear and detailed understanding of how data is distributed across different categories, allowing for a more precise evaluation of the dataset's composition.

The table 4.13 below presents the usage percentages of the pieces extracted from each category. Simply put, it shows the percentage of how much Papadiamantis writes in Katharevousa, AG, and SMG in both the dialogic and narrative parts of his short stories and novels.

Table 4.13: Classification results of Greek language texts across different categories and types.

	Short Stories		Novels	
Category	**Dialogues (DD)**	**Narratives (DN)**	**Dialogues (Md)**	**Narratives (MNar)**
ag	1.802245	2.745226	1.898519	4.695986
kat	65.383727	79.801678	82.140855	87.356234
smg	32.814029	17.453096	15.960626	7.947780

It is observed that Katharevousa shows the highest percentages in each category. However, a significant observation is that Katharevousa's percentages drop to 65.383725 when it comes to the dialogic parts of the short stories, where the percentages of SMG rise, reaching 32.814028. The same rates for the narrative parts of Katharevousa are around 80%, with SMG falling below

[22]Lowercasing, punctuation removal, and whitespace removal.

20%. In comparison with the novels, both in their dialogic and narrative parts, Katharevousa shows values above 80%, although the dialogic parts also display a slightly lower percentage.

4.9 A Further Analysis of the Percentages of Dialogic Parts

Subsequently, in order to further examine points that are considered to contain dialogic parts with greater accuracy, an effort was made to identify patterns in the already isolated dialogic parts that were previously examined, to check if the percentages would remain the same. Specifically, an attempt was made to identify narrative parts, which were collected along with the dialogic ones, and which, however, refer to the dialogue as can be seen in table 4.14:

Example
Εμορμύρισε γελών ακουσίως ο παλούκας
Επρότεινεν ο καλοειδής
Επρόφερε μετά πόνου ο Αντώνης
Έκραξεν ο Ιωάννης Μούχρας πλησιάζων
Απήντησεν ο έτερος των ερετών

Table 4.14: Examples of narrative parts referred to in dialogue, which were collected and measured as dialogic parts

Based on these points, since the punctuation of the prototype text was not helpful in isolating these parts, as it does not contain any systematic pattern to distinguish them from the dialogue (e.g., narrative parts referred to in the dialogue only follow a comma), commas were used to distinguish the two environments that appear, in order to locate them with regular expressions, and to remove them. The first environment is where in a line of text the narrative speech referring to the dialogue follows as shown in table 4.15:

Examples of Narrative Speech Following Dialogue
την κόρην μου θέλω, έκραξε πνέων λύσσαν ο ατυχής πατήρ
κ ημάς που θα μας αφήσετε, έκραξε με δάκρυα εις τους οφθαλμούς

Table 4.15: Examples of Narrative Speech Following Dialogue

While the second environment is where the narrative speech that refers to the dialogue is interspersed within the dialogue, it appears according to the data in Table 4.16:

In this context, for both the first and the second cases, the environments in which this type of narrative speech occurs were identified, through the systematic identification of the verbs

Examples of Narrative Speech Interspersed Within Dialogue
είμαι μισοπνιγμένος, είπε μορμυρίζων ούτος, αλλά δεν είναι τίποτε
όχι πούντς όχι, είπε διά πεπνιγμένης φωνής, κρασί δώστέ μου

Table 4.16: Examples of Narrative Speech Interspersed Within Dialogue

that introduce it. To the best of my knowledge, the verbs that appear in these environments in Papadiamantis's dialogic sections and introduce narrative speech related to the dialogue are available in Github [23].

These environments needed to be distinguished in some way. They could be found interspersed either within or at the end of the dialogue. Thus, commas were used either before the verb or by placing the narrative phrase between commas. Then, regular expressions were applied to remove those portions marked as narrative from the dialogic parts. After applying these regular expressions and removing the remaining narrative sections within the collected dialogic parts, the dialogic parts of the novels and short stories are presented in Table 4.17:

Category	**Word Count**
Dialogic Parts - Novels	34,346
Dialogic Parts - Short Stories	31,894

Table 4.17: Word count of dialogic parts after applying regular expressions to remove narrative elements.

The table 4.18 below presents the new percentages of language categories for the dialogic parts, following the further removal of narrative elements within the dialogic sections:

Table 4.18: Percentage of Dialogic Parts Based on Length for Short Stories and Novels

Type	**Predicted Label**	**Percentage**
	Short Stories	
	AG	2.020196%
	KAT	58.795125%
	SMG	39.184679%
	Novels	
	AG	1.969706%
	KAT	79.456378%
	SMG	18.573915%

[23] https://github.com/dimitrispapad/Papadiamantis/tree/main

Now, the picture of the linguistic categories appears different, though it does not overturn the dominant category, namely Katharevousa. However, in short stories the use of SMG increased by 7% and in novels by 3%, with the consequent drop in the use of Katharevousa. We consider the above table 4.18 to be more representative of the dialogic parts. However, it would not be methodologically appropriate to return those narrative parts removed from the dialogue parts to the narrative parts percentages, as they are organic parts linked to the dialogue, in which case the results of the narrative parts in the table 4.13 remain as outcomes, while the results of the dialogic parts are updated with the table 4.18. The intuition behind this further analysis has been proven to be correct, but the percentages did not highlight the modern Greek as dominant.

4.9.1 Error Analysis and Uncertainty Evaluation in Algorithmic Predictions

An error analysis of the categorization results of Papadiamantis' texts by the algorithm would require a review of each phrase's predictions and the overall percentages by a team of experts in identifying these linguistic categories (linguists, specialists in Modern Greek philology). However, considering the cost of such an undertaking, this thesis opts for a different approach. The categorized data are open access and thus available to anyone willing to undertake such a task. Instead, the present study performs an error analysis that correlates the algorithm's uncertainty with predictions in which the most probable linguistic categories were not clearly dominated by a single option, but rather by two categories with similarly high probabilities for the given phrases presented to the trained algorithm.

Here, the focus is mainly on cases derived from the dialogic parts, where it was hypothesized that SMG should appear in higher proportions. Thus, among the two most probable categories, we observe, in an example from Papadiamantis' text, how the algorithm makes its final decision by selecting the category to which it assigns the highest probability.

Text	**Top Label 1**	**Top Prob 1**	**Top Label 2**	**Top Prob 2**	**Length**	**Decision**
εις τον ποταμόν	kat	0.881	ag	0.119	15	kat

Table 4.19: Example data table with predictions and decisions

However, based on the results of the preceding categorization, cases were identified in which the algorithm assigned high probabilities for a given line of text to belong to two categories. These phrases predominantly fall into either SMG or Katharevousa, and we believe that even if

they were annotated by native speakers of SMG and domain experts in Katharevousa, achieving consensus would be challenging. This difficulty arises in determining when something is entirely Katharevousa, when it is SMG, and when there is an intersection. This intersection is inherent to the nature of Katharevousa itself, as demonstrated by the algorithm's training and the discussion surrounding the nature of Katharevousa. Let us examine some cases where the algorithm shows evident uncertainty, assigning high probabilities to multiple categories:

Text	**Top Label 1**	**Top Prob 1**	**Top Label 2**	**Top Prob 2**
είναι αληθές	kat	0.544	smg	0.453
πιέ να το ξεχάσης	kat	0.507	smg	0.492
εννιά και δεκαπέντε μού χρωστούσεν ο μακαρίτης ο άντρας σου	smg	0.529	kat	0.461
βλάκας εις τας αθήνας κομίζει	kat	0.516	ag	0.484
πως δεν ήρθε μαθές πως δεν ήρθε μαθές	smg	0.508	kat	0.491

Table 4.20: Examples of classifications with the two highest probabilities.

The above indicative cases show that the probabilities of the two most dominant categories are quite close, simply because they combine linguistic elements from at least two categories. At the same time, we know that the algorithm learns by memorizing words, bigrams, and trigrams during its training. For example, referring to the previous table 4.20 of cases with high probabilities for two categories, the first case is a third-declension adjective used in both Katharevousa and SMG. In the second case, we see verbs with verb endings characteristic of Katharevousa, but the syntactic structure is very close to SMG. In the third case, although the structure and vocabulary belong to SMG, the Katharevousa ending *-en* influences the algorithm, while cases like the forth, with syntactic structures from AG but vocabulary from Katharevousa, can again cause uncertainty. What is significant from this investigation of decisions is the fact that we can observe how confident the algorithm is in its decisions and measure its certainty or confidence, in order to reassess the classifications so far within a framework that also accounts for uncertainty.

To investigate this phenomenon, an experiment was conducted using the threshold that the algorithm employs to decide which category to assign to each line.

What is the threshold? The threshold is the point at which the model decides whether a prediction belongs to one category or remains '*uncertain*' In the case we are examining, which pertains to analyzing Papadiamantis' language, the threshold refers to the difference in probabilities between the two most probable categories. Thus, we set a limit between the

two most probable linguistic categories, and if the difference between the highest probabilities (subtracting the second from the first) is greater than the threshold, the prediction is considered certain; otherwise, it is uncertain. For example, if we set a low threshold like 0.05, then in the first example of the previous table, the prediction would result in Katharevousa (0.091), whereas in the second example, where the subtraction yields a result of 0.017, a combination of Katharevousa and SMG would result. This better reflects the reality of linguistic categories.

For our case, in the dialogic parts, the most uncertain instances are considered to be a combination of two categories, with the prevailing combination being that of Katharevousa and SMG. However, this percentage, with a low threshold (indicating that the categories are indeed competing), is particularly low in the final percentages, as shown in the tables 4.21 and 4.22 below. Despite this, it improves the evaluation of certain instances in the dialogic parts of both short stories and novels. The total percentage of uncertain categories for the dialogic parts of short stories now amounts to 1.31 %, while for novels, it is 1.04 %.

Table 4.21: Category Percentages for Dialogic Parts of Short Stories (Normalized by Length)

Category	**Percentage (%)**
kat	58.164984
smg	38.604634
ag	1.917587
kat-smg	0.580648
smg-kat	0.501578
ag-kat	0.085709
smg-ag	0.078466
kat-ag	0.049494
ag-smg	0.016900

Table 4.22: Category Percentages for Dialogic Parts of Novels (Normalized by Length)

Category	**Percentage (%)**
kat	78.971544
smg	18.192665
ag	1.798875
kat-smg	0.329188
smg-kat	0.337865
ag-kat	0.157273
kat-ag	0.155646
smg-ag	0.043386
ag-smg	0.013558

4.10 Discussion

In light of the results presented above, we now correlate them with the views expressed in the literature regarding the two initial research questions of this chapter, namely, in which language does Papadiamantis write? SMG, Katharevousa, or AG? Does the linguistic category he uses differ between the narrative and dialogic parts? An additional third question emerged from observing the results: Is there a linguistic, and therefore possibly a chronological and even genre-based differentiation, between his short stories and novels?

From the results obtained and based on the criteria set for this particular claim in our analysis, we can initially assert that Katharevousa is the language in which Alexandros Papadiamantis primarily writes, both in his short stories and novels. However, the differentiation observed between the dialogic and narrative parts is particularly interesting, with the percentage of SMG consistently increasing in the former compared to Katharevousa. Significantly, there is a noticeable increase in SMG in the dialogic parts of the short stories (about 40%), while in the dialogic parts of the novels, SMG also rises to about 20%. This suggests that in the dialogic parts, the percentage of SMG increases while that of Katharevousa decreases, in both novels and short stories. Lastly, concerning the final research question, it appears that the use of Katharevousa in novels, which remains around 80–90 %, drops to as low as about 60–80% in the short stories, indicating both genre-specific and chronological changes in the linguistic category used by Papadiamantis. Furthermore, the range of variation in the dialogic parts of the short stories (60–80%) is much broader than that of the novels (80–90%), indicating a more consistent use of Katharevousa in the latter.

Therefore, based on the above and the preceding discussion of the literature, this chapter's contribution supports the analysis that Papadiamantis does not write in one clear-cut language but rather utilizes elements from both AG and SMG, as well as Katharevousa. Furthermore, the evidence supports several findings: 1) the dialogic parts exhibit a more systematic use of SMG than the narrative parts both for novels and short stories, but we cannot certainly speak of a dominance of SMG in the dialogic parts, as the best performance was in the dialogic sections of the short stories which reached 40%; 2) the novels contain a higher percentage of Katharevousa compared to the SMG, unlike the short stories where the use of SMG increases; and generally, 3) Katharevousa appears to be the language with the dominant percentage of use. Finally, 4) the usage of AG is certainly very low to zero.

The error analysis of the categorisation by the algorithm of the dataset of the three language

categories showed us the logical confusion of the algorithm (in a small percentage, of course) regarding the Katharevousa, which is a set that intersects between ancient and modern Greek. However, the algorithm recognizes the elements of Katharevousa that survive in SMG, and the elements of SMG and AG that survive in Katharevousa, categorizing them correctly, even though they are in a different category from their actual one. These are cases where the algorithm works correctly, assigning an '*incorrect label*', as shown in Table 4.6. Finally, the error analysis regarding the uncertainty of the algorithm in the predictions of Papadiamantis texts, despite the fact that it identifies the combination of uncertainty in the categorisation of Katharevousa-SMG as the first candidate, remains at a very low rate for both dialogic and narrative parts.

Chapter 5

Identifying Topics in Papadiamanti's Short Stories

5.1 Introduction

Many studies of Papadiamantis' work (as well as that of other authors) in the context of traditional literary criticism focus on highlighting the particular thematic motifs that permeate specific short stories or novels, or an especially prominent narrative, or even shared thematic patterns across different writers. Undoubtedly, within the framework of conventional literary analysis, collecting and highlighting all thematic motifs across Papadiamantis' short stories would be a challenging endeavor and perhaps beyond the scope or interest of a close reading approach.

On the other hand, from the perspective of computational literary criticism, the ability to detect and highlight topics in the work of Papadiamantis (and in the work of other authors as well) appears to be a challenging task, whose difficulties and benefits will be discussed in this chapter.

5.2 Related Work

To the best of our knowledge, no previous study has attempted to identify computationally derived topics in the corpus of Alexandros Papadiamantis. In traditional literary analysis to date, scholars have tended to highlight a specific thematic aspect of the Papadiamantis corpus by selecting those texts and pieces of evidence that support it. Indicatively, we note a few works that

bring to light certain thematic motifs in Papadiamantis' writings or offer interpretive analyses of themes in specific works: on religiosity in Papadiamantis Άγρας (1934), Παπαϊωάννου (2005), Τριανταφυλλόπουλος (2011), and Τωμαδάκης (2005); on humor Δάφνης (2005); on the comic element Καμπατζά (2011); on Skiathos and its relationship to Papadiamantis' work Merlier (2005); on connecting *The Murderess* with a Darwinian framework Politi (2005); on connections with dark fairy tales Πασχάλης (2001); on Papadiamantis' relationship with the traditional romantic tradition Constantinides (1997); for the hidden ecclesiastical tradition in the murderess Χελιδώνη (2010), for Papadiamantis' relationship with Russian formalism Παπαγιώργης (1997), among many other interpretations. The list of interpretative analyses of aspects of Alexandros Papadiamantis's work, both short stories and novels, is particularly long. For a more extensive comparison of analyses, see the critical edition by Φαρίνου-Μαλαματάρη (2005), which includes a set of important interpretative approaches.

The main distinction of the present chapter from previous thematic studies is that it seeks to identify topics computationally in the corpus of all Papadiamantis' short stories. The central question guiding this analysis is therefore the following: Can we bring together all of Papadiamantis' short stories, apply topic modeling, and ultimately detect distinct and salient themes within his short story corpus? However, it is first needful to discuss how models can represent the meaning of words, and ultimately a broader meaning of the text, starting with basic concepts in lexical semantics, the usefulness of this type of analysis in the field of literary studies, and finally the way in which the models themselves work.

5.3 Lexical semantics

Understanding the meaning of a word lies at the heart of semantics and logic (Saeed, 2015), as well as Natural Language Processing (NLP). In the tradition of formal semantics, lexical meaning is often treated compositionally, with frameworks such as Montague Semantics (Montague, 1973, 1974) aiming to show that natural language can be analysed with the same rigour as formal logic. Building on this, Dowty (1979) introduced lexical semantics into Montague's framework, proposing that verbs, for instance, can be represented through smaller semantic components such as CAUSE, BECOME, and NOT(alive(x)). These rule-based approaches allow precise interpretation but are not typically used in computational models, as they require extensive manual rule construction and do not scale well across new vocabularies or languages.

Instead, an empirical solution to the computational problem of understanding word meaning comes from the *Distributional Hypothesis* (Firth, 1957b; Harris, 1954), which holds that a word's meaning is reflected in the company it keeps. That is, we can learn something about a word's meaning from the words it frequently appears with. For example, while *murderer* and *weapon* are semantically different in type, one denoting a person, the other an object, their frequent co-occurrence suggests a strong relationship (Jurafsky & Martin, 2024). Based on this principle, modern NLP approaches represent words as vectors in a multidimensional space, capturing these co-occurrence patterns. Models such as Word2vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014), and BERT (Devlin et al., 2019) all build on this idea, offering different techniques for learning these word representations from large corpora.

5.4 Vector semantics

Before we discuss vector semantics, it is crucial to provide a definition of a vector. What is a vector? A vector is an array of numbers, which are arranged in sequence, and we can identify each number by its index (Goodfellow et al., 2016). It is also important to define what a vector contains, but its typical form is as follows:

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}.$$

Figure 5.1: Representation of the vector x as a column matrix, consisting of elements $x_1, x_2, \ldots, x_n$ Goodfellow et al. (2016).

Let's also keep in mind; vectors can be viewed in linear algebra as points in space, with each element providing the coordinate along a different axis (Goodfellow et al., 2016).

Building on this fundamental concept, vector semantics is the primary method for representing the meaning of a word in NLP, allowing us to outline many aspects of a word's meaning. The historical and ideological roots are found in the merging of two major ideas in the 1950s. This merger involves the idea of Osgood (1957), who proposed using a point in three-dimensional space to represent a word's co-reference, and the proposals by linguists Joos (1950), Harris

(1954), and Firth (1957a), who suggested defining the meaning of a word from its distribution in linguistic use, that is, from the words neighboring it or the grammatical environments. Thus, within these frameworks, we assume that words appearing in similar contexts tend to have similar meanings. This basic idea, which connects how words are distributed to what they mean, is called ' *the distributional hypothesis* ' (Schmidt, 2015), as outlined earlier in this section.

Now, the basic idea of vector semantics is to represent a word as a point in a multidimensional (semantic) space, which is generated from the distribution of neighboring words. Let us see how words can be represented as vectors using an example from a sentiment analysis task which represents closely related words in a multidimensional space by Li et al. (2015), where words are represented as vectors from specific linguistic categories (Visualization on latent representations for modifications and negations):

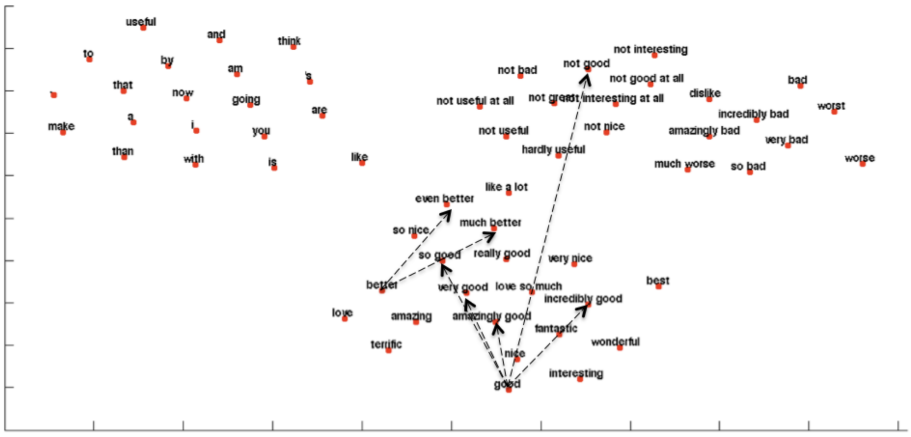


Figure 5.2: Visualization on latent representations for modifications and negations

Generally, vectors representing words are called embeddings, while the term embeddings is derived from the mathematical concept of mapping from one space or structure to another. However, the term is more closely associated with dense vectors (Jurafsky & Martin, 2024), and thus in this work, whenever reference is made to embeddings, it will refer to dense vectors. Now, what does it mean to represent a word as a vector? The concept involves assigning each word a number of scores that position it within an arbitrary space. Unlike just one or two dimensions, which do not permit many differing relationships, this space encompasses hundreds (Schmidt, 2015). Indeed, representing this multidimensional space is challenging precisely for this reason. Two very popular methods in NLP for performing the task of representing words in a multidimensional space and the resulting semantic relationship of words are TF-IDF (term frequency-inverse document frequency) and word embeddings. In this work, word embeddings

are used; however, it would be an omission not to briefly mention TF-IDF and its disadvantages compared to word embeddings.

5.4.1 TF-IDF (term frequency-inverse document frequency)

TF-IDF is a statistical method that evaluates the importance of a word in a given text, combining two key metrics: the frequency of the word's appearance in the specific text under examination, and how rarely the word appears across a collection of texts (Jurafsky & Martin, 2024). Thus, articles, prepositions, and generally a set of frequently occurring functional words are de-emphasized when examined more comprehensively. This method had a series of issues that word embeddings address more satisfactorily: TF-IDF represents words based solely on their frequency of occurrence and not any semantic content, and creates sparse representations, whereas word embeddings create dense vectors. The significance of this, is particularly important. Sparse representations are vectors where most values are zero, while dense representations use fewer dimensions, and most values are non-zero, including negative values, thus offering greater computational efficiency and precision in representing semantic relationships (Chatzkyriakidis, 2024). For example, if we wanted to represent the appearance of the word *mouse* in a 1000-word text, in the sparse representation it could be represented as a 1000-dimensional vector, with unique non-zero values where the word *mouse* appears [0,0,0,0,1,0,0,0...]. In contrast, in the dense vector of word embeddings, much fewer representations with zero values would be used [0.6, 0.1, -0.2...].

5.4.2 Word Embeddings

Let's then delve into word embeddings, the short dense vectors. The first and fundamental question is how word embeddings are defined and what exactly they are.' *Word embeddings are dense, distributed, fixed-length word vectors, built using word co-occurrence statistics as per the distributional hypothesis* ' (Almeida & Xexéo, 2019; Blacoe & Lapata, 2012; Schnabel et al., 2015; Turian et al., 2010), or, put it more simply, ' *word embeddings are real-valued vector representations of discrete words* '(Mikolov, 2013; Mikolov et al., 2013). As mentioned previously, the vectors of embeddings are small, ranging between 50-1000 dimensions [1], unlike

[1]Although we will see in detail the training of a word embedding package, for now, it suffices to say that the way each word's score and dimensions are learned is as follows: 'We train a model that 'learns' scores for each word in the text for some arbitrary number of characteristics. The number of characteristics that we choose are called the dimensions: each word occupies a distinct point in that broader space (Schmidt, 2015).

the huge number of vectors related to the size of the vocabulary in other approaches. Jurafsky and Martin (2024) mention that although we do not fully understand why dense vectors perform better than sparse vectors, there are some basic intuitions behind this. Firstly, representing words as 300-dimensional vectors simplifies the learning process for classifiers (especially when considered in relation to the large size of the entire vocabulary or a corpus of texts that the TF-IDF method would use), while the fewer number of parameters enhances their ability to generalize, and they perform exceptionally well with cases of synonymous words (Jurafsky & Martin, 2024).

Word embeddings are particularly effective in capturing analogical relationships between words. (Chatzkyriakidis, 2024; Schmidt, 2015). Actually, this is one of the great advantages of word embeddings. Through this analogy, we can see semantic relationships between words. What do we mean by the notion of analogy? We can understand analogy by considering the classic example where the following operation with the word vectors (king, man, woman, and queen) would yield the following result: vector(king) - vector(man) + vector(woman) is close to vector(queen) (Schmidt, 2015). In addition, we are in a position to identify polysemy [2] as well as broader relationships between words. Finally, looking at the big picture of word embeddings, which can be a standalone feature in NLP tasks (Turian et al., 2010), it is worth noting the fact that they encode surprisingly accurate syntactic and semantic word relationships (Mikolov, 2013).

5.5 Why Word Embedding Models in Digital Humanities?

Now, in connecting word embeddings with the humanities, the fundamental question is: Why should we use word embeddings in digital humanities? These models, word embedding models, may hold nearly as many possibilities for digital humanists modeling texts as do topic models (Schmidt, 2015). More specifically, word embedding models provide a spatial analogy to relationships between words [3]. That is, they take an entire corpus, and try to encode various relations between words into a spatial analogue [4]. In reality, they try to ignore information to

[2]Polysemy is the existence of multiple meanings for the same word (Schmidt, 2015).

[3]Apparently this approach is, as we said earlier, within a linguistic framework in which the view is adopted that the distribution and meaning of words are linked (Harris, 1954), and we can represent through these models the inter-relationships between words.

[4]This is, in fact, the key advantage of this approach compared to tools like Voyant Tools (https://voyant-tools.org), which, while representing word frequency within a corpus along with the surrounding context of each word's occurrence, do not provide the capability for a quantifiable comparison of words based on their appearance

focus on the relationships between words.

What questions does a word embedding model pose? What do we attempt to see through its use in digital humanities, especially in the texts of Alexandros Papadiamantis? Essentially, a word embedding model asks: what if we could model all relationships between words as spatial, or in other words, how can we fit words into a field defined by the relationships between them. This question posed by the word embedding model is particularly useful for research in digital humanities more broadly, as it allows us first to see words that are similar to each other and to learn something broader about the texts from their relationships. The two main goals of these word embeddings models are that they attempt to represent the similarity of use between words in space, and the relationships between words and their similar paths in space. Thus, one possible way to use them in the digital humanities so that they truly have something to tell us is by being able to identify correlations of words, themes, and common places within the text (Piper, 2019; Schmidt, 2015), and more specifically within the texts of Alexandros Papadiamantis.

In fact, what will be attempted using these models is a topic modeling approach to Papadiamantis' short stories. However, before moving on to how this approach has been applied in this chapter, let us first answer the following questions: 1) what is a topic? 2) what is a topic model? 3) what is the significance of topic modeling in the digital humanities? and 4) how can it be implemented in the digital humanities?

So first of all, let us give a formal definition of a topic, in a topic modeling context [5]. '*Topic is a distribution over a fixed vocabulary of words*' (Blei, 2012), reflecting either a semantic correlation between words or meaningful connections shaped by the unique characteristics of the specific (literary) text being analyzed (e.g., genre, author, historical period, etc.). For example, if we aimed to represent the topic of family in terms of semantic correlation, we could envision a model applied to a text, returning highly similar words such as mother, father, sister, brother, etc. However, if we sought to represent the theme of love in Shakespearean texts,

in different contexts using measurable terms. Unlike Voyant, which simply visualizes individual occurrences, word embeddings allow for vectorization of each word based on its contextual appearances, vector comparison, visualization of word relationships in a two-dimensional space, and training of a model that identifies semantic relationships between words. Or, put more simply, the only way to perform this kind of comparison using Voyant Tools would be manually, with all the limitations and costs that this entails.

[5]This definition is adopted in most topic modeling approaches using conventional models such as Latent Dirichlet Allocation (LDA) (Blei, 2012) and Non-Negative Matrix Factorization (NMF) (Févotte & Idier, 2011), as well as more recent models like BERTopic (Grootendorst, 2022) or Top2Vec, which leverages Doc2Vec's word and document representations to jointly learn embedded topic, document, and word vectors (Angelov, 2020; Le & Mikolov, 2014)

Romeo and Juliet would be associated with this theme, not through direct semantic similarity, but through meaningful connections between these words (characters) and the concept of love. Thus, in conducting our interpretation, it is crucial to acknowledge that in the literary domain, relationships between words can be identified based on semantic similarity but also through literature-specific connections, which extend beyond conventional similarity measures, as in the example of family-related terms.

By topics in computational criticism, of course, we do not mean that the model is able, as in traditional literary criticism, to highlight a selected aspect, a thematic motif of Papadiamantis's work extensively with arguments. Instead, lists of words are identified as thematic patterns. Simplification is in a sense a main cost of computational hermeneutics in order to understand complexity on a large scale (Piper, 2015), or in a positive view, the best way to understand a text is to change it, to transform it, an approach that is also followed in traditional literary criticism (Ramsay, 2011).

Piper (2019) poses the question of what to do with these lists of words in the context of computational criticism. He explores the topics [6] by associating it with the greek word *topoi* (common places) or with the latin expression *locus communis*. As he characteristically writes '*a topic is a generalized associational pattern*' defining them with respect to plot and linearity in narrative as '*semantic units that are more temporally invariant and pull us out of time and into the realm of space, shape and form*'.

Piper (2019) argues that through the identification of topics we can see the texts topologically, so that we can interpret the relationships between topics. He is interested in how a topic is differentiated within a set of texts, but also within a set of topics. His perspective focuses on the differentiation, the heterogeneity of a topic within the dataset under consideration, when the different semantic fields are formed. Also, he states that there is a co-constructibility in the creation of the computational topics, in the sense, that the observer's position, interests, interpretive actions, the dataset, and the model itself are part of our own modeled construction.

The core of his analysis lies in the type of topic models he employs. But what is topic modelling? Topic modelling is an unsupervised task in NLP aimed at uncovering latent themes in a collection of texts. Traditionally, this has been approached through probabilistic models such as Latent Dirichlet Allocation (LDA) (Blei, 2012), where topics are defined as distributions over words, and each document is viewed as a mixture of those topics. This approach is based on word

[6] Piper (2019) uses mostly the term '*topics*' interchangeably with terms '*topoi*'.

co-occurrence patterns, following the intuition that words that frequently appear together likely belong to the same thematic cluster. LDA has become particularly popular due to its simplicity, interpretability, and scalability, making it an effective tool for analysing large unlabelled text corpora across diverse fields, including literary studies.

However, in Piper's view of topics in computational criticism, we believe that it is organized and based on probabilistic models, making it a highly method-driven approach. By this statement we mean that if we could come up with another method for identifying computational topics beyond probabilistic models, perhaps a fresh perspective could be given on what topics are in computational criticism and how they organize the language of the literary text. In his analysis, Piper (2019) chooses frequency matrices to generate embeddings while excluding the use of embedding models to represent words in semantic space for two reasons. The first is that he has a small number of texts available to him in his study (203,348 words), so he does not expect these models to produce strong results, and the second is because he claims to be interested in how terms are related to each other in the texts under consideration, rather than in understanding their overall context.

Differentiating our perspective in this thesis, we will attempt to examine the potential contribution of these word embedding models to perform a topic modeling approach. Our aim is not to follow a trajectory where the same topics are merely distinguished within given texts, but rather to identify the underlying topics of a work or a body of work. Thus, from our perspective, topic differentiation can be understood through the varying associations of topic words within each clustering, while the broader contexts help to reveal strong thematic aspects of the text, as explained below.

However, what is important in this approach is that we use the optimally generated results of the models, which are identified via a grid search method (i.e., a systematic search over parameter combinations), with feedback from the clustering technique that follows. In particular, by evaluating the quality of the clusters generated by the word embedding models using metrics, we find that it is possible to identify thematic correlations that emerge as distinct and coherent within a word embedding model trained as effectively as possible.

Thus, we are not concerned with the different correlations of a single topic. Instead, we focus on all the distinct and coherent correlations that emerge after training the word embedding models and applying the clustering algorithm. In this way, we aim to identify prominent topic lines within Papadiamantis' topic space.In this sense, this method is called upon to contribute to

topic modeling within the analysis of literature, in science-for-science contexts, by comparing the finding of salient thematic correlations in a series of texts against the thematic differentiation and specificity of a particular topic. In doing so, it brings into focus the most prominent topic correlations in Papadiamantis's work.

To give an example, Piper (2019) is interested in how the thematic word *heart* is associated differently across thematic patterns in a nineteenth-century poetry collection, in order to demonstrate the internal variation of a topic by showing how it is shaped across distinct themes that contain it.

Examples of Topics Centered Around 'Heart'

Topic 5: tear, heart, sorrow, grief

Topic 46: love, heart, kiss, tender, passion

Topic 48: dread, fear, heart, suffer

Topic 96: happy, life, hope, heart, joy

In our own approach, the fact that specific words are grouped together, and these groupings are evaluated by metrics (which we will explain in detail below) may suggest that the resulting clusters occupy their own semantic field in relation to the rest of the corpus. This, in itself, is an interpretable phenomenon, both in relation to Papadiamantis and to any author or text. That is, if we can locate thematic fields within the semantic space of a text, this may allow certain topics to surface more clearly, defining not only their mutual relationships but even the criteria for their generation. This allows us to see topics across the entire dataset under consideration, but also to interpret the topic itself based on the words associated with it.

An example of a prominent thematic space in Papadiamantis, one suitable for interpretation based on both its internal associations and the fact that it forms a coherent semantic region in the model, could be the following:

Cluster: μήτηρ (mother), γυνή (woman), σπίτι (house), παιδιά (children), σύζυγος (husband)

So, if these words are included in the same cluster, it would imply that 1) they have enough correlation with each other to be clustered and 2) they have sufficient differentiation from the other words to be grouped together. Based on these two deductions, which follow from the application of clustering, we can argue that this is a broader semantic correlation (words that are close together in the multidimensional semantic space), which may also reflect the thematic correlation (representation of a woman) that can be interpreted by the relationships between its members (a topic in which women are represented in their roles as mothers and wives, with a constant presence in the home).

To sum up, exploring the fundamental need in literary criticism to highlight and interpret topics, drawing on the essential positivist perspective in Piper (2019) concerning the search, interpretation, and identification of topics in computational models, we propose an approach to topic search and interpretation that leverages word embeddings and clustering methods in order to identify strong thematic axes in the topic space of a text corpus, or in Piper (2019) terms, to establish semantic maps in common places. Of course, being aware of the limitations of this thesis, a key computational issue is that embeddings belong to the category of static rather than contextual, so this remains a limitation but also an opportunity for future study.

5.6 Word2vec

An easy-to-train, accessible, and fast method for performing NLP tasks, particularly for training word embedding models, is Word2vec, a technique for obtaining word embeddings [7]. Wor2vec is not a standalone algorithm but it is a family of model architectures and optimizations that can be used to learn word embeddings from large datasets as mentioned in the TensorFlow website [8]. Word2vec provides static embeddings, which means that it learns a fixed representation (embedding) for each word in the vocabulary. But how Word2vec is trained in order to represent word as embeddings? In fact, using a corpus, word embeddings are trained by modifying vector representations in response to how well the model predicts target or context words. This package provides an efficient implementation of the Continuous Bag-Of-Words (CBOW) and Skip-gram architectures for computing vector representations of words. The intuition behind Word2vec is that instead of counting how often each word appears near the word of interest, a classifier will be trained on a binary prediction task which predicts whether it is likely that a word will

[7] https://code.google.com/archive/p/Word2vec/
[8] https://www.tensorflow.org/text/tutorials/Word2vec

appear near the word of interest. However, the actual goal is not the prediction task itself, but learning the weights from the algorithm (i.e., numerical values that capture how strongly words are related), as word embeddings (Jurafsky & Martin, 2024).

In relation to what we discussed in the previous chapter 4 about *supervised* ML algorithms, in this case, the algorithm is *self-supervised*, and it learns the gold labels without us providing them (Goodfellow et al., 2016; Jurafsky & Martin, 2024). For example, if the word w appears near a word of interest, then this is a correct response to the question posed, to predict the most likely words to appear near the word of interest.

The main learning algorithm is a Skip-gram with Negative Sampling (SGNS), with the basic intuition of treating each target word and its neighboring context as positive examples. Initially, the algorithm randomly samples from other words in the vocabulary to obtain negative examples. Subsequently, it employs logistic regression to train the classifier to distinguish between these cases. Finally, it utilizes the already learned weights as embeddings. For example, in SGNS, if we have a sentence like '*The dog sits in the house*', the model takes the target word as an input (E.g. '*dog*') and aims to predict the context words ('*The*', '*sits*' '*in*', '*the*', '*house*').

The other algorithm of Word2vec is the Continuous Bag of Words (CBOW). In the CBOW architecture, an attempt is made to predict a specific word from a set of surrounding context words. Essentially, the model takes the surrounding words as input and tries to predict the central word. For example, if we have the sentence '*The cat sits in the house*', and we want to predict the word '*cat*' using the words '*The*', '*sits*', '*in*', *the*', '*house*' as context, the CBOW would attempt to identify the word '*cat*' based on these words (Mikolov, 2013). Overall, the main difference between CBOW and SGNS lies in the loss function used to update the model. Specifically, while CBOW trains a model that attempts to predict the target word from its context, SGNS does the opposite, the target word is used to predict each word in its context (Mikolov, 2013). This architecture of Word2vec makes it much simpler than a neural network (training a logistic regression classifier instead of a multilayer neural network with hidden layers that require more sophisticated training algorithms), along with the fact that it simply has the task of binary classification, and not prediction of every next word (Jurafsky & Martin, 2024; Schmidt, 2015). The two different architectures of Skip-gram and Continuous Bag of Words are represented in the following figure:

However, how is the algorithm trained for the classification task [9]? Let us assume that we

[9]Below, we focus the discussion on the Skip-gram algorithm as it is considered the primary algorithm of Word2vec, and it is also the one used in the current work

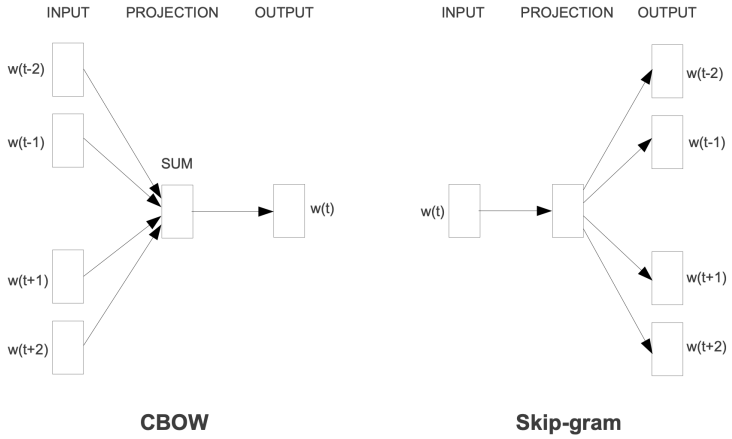


Figure 5.3: The architecture of Skip-gram and Continuous Bag of Words (Mikolov, 2013)

have a context window (i.e., the range of context the algorithm looks at includes the two previous and the two next words) of ± 2. The main goal is to train the classifier such that, given a tuple of a target word w combined with a candidate context word c, it will return the probability that the candidate word c is a true context word ($P(+|w, c)$). On the other hand, the opposite probability, where c is not a true context word for w, is simply $1 - P(+|w, c)$ ($P(|w, c) = 1P(+|w, c)$) (Jurafsky & Martin, 2024).

How does the skip-gram algorithm compute this probability? The intuition is to base this probability on embedding similarity. That is, a word is likely to appear close to the target if its embedding vector is similar to the neighboring embedding. To calculate the similarity between these two (dense) embeddings, we consider two vectors to be similar if they have a high dot product (Jurafsky & Martin, 2024) [10].

However, the result of this multiplication will be a number from negative infinity to positive infinity, not a probability. The multiplication of these two vectors and their dot product comes from linear algebra, and although it is written in this way, we actually mean the sum of the products of the corresponding elements from each vector. For example, if we represent a random

[10]The following operation can calculate the dot product between two vectors v and w, where we multiply their corresponding elements one by one in sequence and then sum them up. The dot product of two vectors, $\mathbf{v}$ and $\mathbf{w}$, denoted as $\mathbf{v} \cdot \mathbf{w}$, is a scalar quantity that results from the summation of the products of their corresponding components. Mathematically, it is expressed as:

$$\mathbf{v} \cdot \mathbf{w} = \sum_{i=1}^{N} v_i w_i$$

where N is the number of dimensions of the vectors, and v_i and w_i are the components of vectors $\mathbf{v}$ and $\mathbf{w}$, respectively (Jurafsky & Martin, 2024).

word x as $[1, 0]$ and another random word y as $[1, 0.5]$, their dot product is $1 \cdot 1 + 0 \cdot 0.5 = 1$, suggesting a high similarity and a higher probability of being contextually related. To convert this number (product) into a probability, we must use the logistic or sigmoid function [11] (or any other function that transforms any value to an interpretable value between 0 and 1) $\sigma(x)$, where:

$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$

This function maps a number to a probability between 0 and 1. [12]

Thus, applying this sigmoid function to the probability of c being a true context word for target w, we would have the following expressions, the first for when c is a context word, and the second for when it is not:

$$P(+|w, c) = \sigma(c \cdot w) = \frac{1}{1 + \exp(-c \cdot w)}$$

$$P(-|w, c) = 1 - P(+|w, c) = \sigma(-c \cdot w) = \frac{1}{1 + \exp(c \cdot w)}$$

In this way, we derive the probability of being a context word, but we know that there are many context words in each context window. Therefore, using the skip-gram model, we make another '*naive*' assumption, similar to that used in Naive Bayes, which states that other context words are independent of each other, allowing us to multiply their probabilities:

$$P(+|w, c_{1:L}) = \prod_{i=1}^{L} \sigma(\mathbf{c_i} \cdot \mathbf{w})$$

Figure 5.4: Independent computation of probabilities for context words

Summing up, the Skip-gram model trains the probabilistic classifier, and given a test target word w and a context window consisting of L words, it assigns probabilities based on how similar these context windows are to the target word. This probability is based on the application of the sigmoid function to the dot product of the embeddings of the target word as well as each context word. Therefore, to compute this probability, we need the embeddings for each target

[11] the two terms are used interchangeably

[12] This sigmoid function belongs to the category of activation functions. In the context of neural networks, an activation function is a mathematical function applied to the output of a neuron, usually to determine its activation level, helping to decide when the output from the neuron will be passed to the next layer of the network. Without the activation function, the network would be a linear function without the ability to model complex relationships (Goodfellow et al., 2016).

word and each context word, which is exactly what the Skip-gram does, storing two embeddings for each word, one for when the word is the target and one for when the word is a context word (Jurafsky & Martin, 2024).

So, the two metrics we need to know are w and c, where each contains an embedding for every word in the vocabulary. The learning algorithm for Skip-gram embeddings takes a text corpus as input and a selected vocabulary size n. Initially, it assigns random embedding vectors to each word n in the vocabulary, and then proceeds with repeated adjustments of the embedded vector of each word w to make it more similar to the embeddings of words that appear near it in the texts, and less similar to the embeddings of words that do not appear near it. Words that appear nearby serve as positive examples, while those that are distant serve as negative ones (which are more numerous).

A particularly important point in the process is the negative examples; we need negative examples (even more than positive ones). A sample with negative examples will contain words that do not appear near a target word and noise words, that is, random words from the vocabulary selected based on the weighted unigram frequency. For example, a word that appears very frequently, such as an article, will have a high probability of being a noise word due to its high frequency of occurrence. The final goals of the training algorithm are thus to maximize the similarity of target words with paired context words through positive examples, and to minimize the similarity of target words with negative context words from the negative examples. In other words, the aim is to maximize the dot product of a word with real context words, and minimize the dot product with negative words, non-neighboring words, and very frequent words (Jurafsky & Martin, 2024).

5.7 fastText

In order to have a comparison with Word2vec, we considered that we should train another model in generating (static) word embeddings for Papadiamantis' texts so that we can compare in our interpretive analysis the results of both models. The model chosen is fastText [13].

fastText is an is an open-source, free library that allows users to learn text representations and text classifiers. Based on our purpose, we focus on the part of training the model to learn text representations by creating embeddings. This model was created in order to generate more

[13] https://fastText.cc

accurate embeddings, overcoming the limitation of models such as Word2vec, which could not capture morphological information of words, essentially assigning as we saw in the Word2vec section 5.6 a separate vector to each word. This limitation may constrain the accurate generation of embeddings especially in languages such as Greek, Turkish, Finnish etc., which have rich morphology , or in corpora that include many rare words and large vocabularies (Bojanowski et al., 2017). The great advantage of this approach is that, in fastText, the representation of vectors is related to the set of characters that make up a word. That is, words are the sum of the character vectors, so that the model can also focus on information within the word.

Let's now examine how the model works, and how it differentiates from and converges with Word2vec in its state-of-the-art form. To begin with, this model is an extension of the continuous skip-gram model we saw in Word2vec (Mikolov, 2013; Mikolov et al., 2013), but it also takes subword information into account. As in Word2vec, given a word in the vocabulary of size W, where the word ID is identified by its index $w \in \{1, \ldots, W\}$, the result we wish to obtain is that the model learns a vector representation for each word, similar to the skip-gram model with negative sampling (Le & Mikolov, 2014; Mikolov, 2013).

However, this model attempts to emphasize the internal structure of words. More specifically, each word is represented as a bag of character n-grams. At the beginning and at the end of each word special symbols are placed to indicate the boundaries of the word such as (<) for the beginning and (>) for the end, enabling prefixes and suffixes to be distinguished from other character sequences. In addition to the set of character n-grams, the representation also includes the complete word itself. Let us look at an example given by the authors in the article in which this model was proposed (Bojanowski et al., 2017). Let's take for example the word '*where*' and n=3. The word before training will be represented as: <wh, whe, her [14], re> and <where>.

Our assumption is that we have a given dictionary of n-grams of size G. For any word w, let $G_w \subset \{1, \ldots, G\}$ be the set of n-grams that appear in w. Each n-gram g has its own vector representation z_g. Informally, the idea here is that instead of learning a vector for the whole word, we build its representation by summing the vectors of the smaller n-grams that it contains. This allows us to handle rare or unseen words more robustly, since their building blocks (n-grams) may still be known. This idea is captured by the following scoring function, which computes the similarity between a word w and a context c based on the sum of the dot

[14]This particular her would be different from her which would be the personal pronoun, as one is a member of the tri-gram while the other is a whole word.

products between the n-gram vectors z_g and the context vector v_c (Bojanowski et al., 2017):

$$s(w, c) = \sum_{g \in G_w} z_g^\top v_c$$

5.8 Applying Word2vec and fastText to the Papadiamantis texts

In this section, we will discuss how Word2vec and fastText were applied to Papadiamantis' short stories. To begin with, it is important to mention that the models were trained with the single corpus of all Papadiamantis' short stories (Dimitroulia, 2021) and the manner of their training is discussed in this subchapter.

Initially, the texts were preprocessed by removing punctuation marks, converting all letters to lowercase, and changing the final ς to σ. Then, in order to remove stopwords, which create noise and hinder the performance of the model due to their high frequency and insignificant contribution to the semantic relationship search of words, a stopword list was created. This list appears to be satisfactory for the texts of Papadiamantis and is provided in this repository on GitHub. Specifically, the stopword list consists of stopwords derived from the NLTK package for Greek, from SpaCy[15], from a stopwords package for Ancient Greek[16], and from the functional words in Papadiamantis' works, that were identified after each model training, during which any stopwords that appeared in the results, were manually added to the list [17] (all words in the stopword list that were in polytonic system were converted to monotonic). Subsequently, a very critical step was that the text was split into sentences using SpaCy's *sent tokenizer* [18], before the training of the models starts.

Finally, both the Word2vec model and fastText were loaded from the Gensim open-source library,[19] and trained on the preprocessed and tokenized sentences, in order to generate word embeddings for each remaining word after the previous processing steps and the application of specific parameters, which will be discussed shortly.

Generally, the literature suggests that word embeddings created with Word2vec perform

[15]https://raw.githubusercontent.com/stopwords-iso/stopwords-el/master/stopwords-el.json
[16]https://github.com/aurelberra/stopwords/tree/master
[17]The list of words included in the stopwords is around 3000 words and is available here:
[18]https://spacy.io/api/tokenizer
[19]Gensim is an open-source library that supports, among other things, models for vector semantics, topic modeling, etc. You can find it here: https://radimrehurek.com/gensim/#

better when they are focused on the structure of a corpus, its unique characteristics, its nature, and theme, rather than when training has been conducted on large topically unconstrained corpora with a global application (Diaz et al., 2016), while our expectation for fastText is similar. Therefore, training Word2vec and fastText on the texts of Papadiamantis leverages the writings of Papadiamantis himself, and consequently, we expect relatively good performance from the models due to the common origin (literary text in the broader sense) and source (a single author) of the data. However, for Papadiamantis, and for each writer considered from a computational criticism perspective, there is no golden mean. This means that the point at which parameters are set correctly to yield true semantic and thematic similarity is determined through experimentation and human examination of the results. The more meaningful the experimental results, the more we consider the model to be improved [20]. Before we move on to how we found the optimal value for each parameter the following diagrams show the parameters that need to be set in each model:

Listing 5.1: Tunable Word2vec configuration

```
model = Word2vec(
    sentences=tokenized_sentences,
    vector_size= ,      # Dimensionality of word embeddings
    window= ,           # Context window size
    min_count= ,        # Minimum word frequency
    epochs= ,           # Number of training iterations
    sg= ,               # Skip-Gram (sg=1), CBOW (sg=0)
    workers= ,          # Number of worker threads
    seed= ,             # Random seed
    negative=           # Number of negative samples
)
```

[20]This could be a distinctive feature of literary analysis through such methods, but it also represents a limitation of the current research. Specifically, if someone, through experimentation, sets the model parameters better than in the current work, they may be able to identify different, and perhaps better, thematic correlations in the texts of Alexandros Papadiamantis.

Listing 5.2: Tunable fastText configuration

```
model = fastText(
    sentences=tokenized_sentences,
    vector_size= ,      # Dimensionality of word embeddings
    window= ,           # Context window size
    min_count= ,        # Minimum word frequency
    sg= ,               # Skip-Gram (sg=1), CBOW (sg=0)
    epochs= ,           # Number of training iterations
    workers=            # Number of worker threads
)
```

The first parameter, `sentences=tokenized_sentences`, provides the tokenized sentences generated by the SpaCy Greek model, as described earlier. Specifically, `tokenized_sentences` is a list of cleaned and tokenized sentences, where each sentence is itself a list of word tokens (strings), which are used as input for training the model. The *vector size* refers to the dimensions of the vectors, and we know that the larger the number, the more detail the model is capable of capturing, but there are risks of overfitting on small datasets and computational cost. The *context window* is a crucial feature. It refers to the number of context examined around each target word (before and after the target word). The next parameter selected is the *min count*. This specific parameter is set to take into account the model words that must appear at least throughout the entire corpus, as much as the minimum number we define. The parameter *epochs* refers to the number of times the model goes through the entire training dataset. We can think of this parameter as the number of times the model reads and rereads the training dataset. More epochs can improve performance, however too many epochs can cause overfitting. Next, the CPU parameter *workers* refers to the number of parallel threads (CPU cores) used for training, but we won't be concerned with it as it is predefined according to the computational capabilities of each computer [21].

In addition, for Word2vev in particular, the *seed* parameter attempts to limit randomness and ensure reproducibility. That is, because each time a model is trained it makes different random choices, such as how the training data is shuffled, which negatives are sampled are picked or which weights are initially assigned randomly. This parameter ensures that the same procedure is followed, so that the results (such as vectors, word similarities) are reproducible. In ML there is already a predefined value for this parameter. Moreover, the *negative* parameter (Goldberg

[21]The above models were executed in a Jupyter Notebook environment using Anaconda, on a MacBook Air with an Apple M1 chip and 8 GB of RAM.

& Levy, 2014), used for negative sampling, and more precisely to specify how many random (negative) words will be sampled for comparison for each word pair. Finally, the use of the Skip-gram parameter was chosen in both models, with *sg*=1, whose training and contribution were previously described, as we expect it to perform well both with rare words and with large datasets compared to CBOW.

Overall, in order to select the optimal performance values of the model, many experiments were conducted with the previous parameters, while the ultimate goal is to find hyperparameters, that are not based on the bias of an evaluator but on some independent metric. We should keep in mind, that the determination of hyperparameters, is task-specific decision, meaning that different problems require different optimal hyperparameter configurations (Mikolov et al., 2013). Thus, it was decided to define the fixed parameters proposed in the literature in proportion to the size of the dataset and the basic function of the model. Since the parameters could be evaluated only through interpretation of the associations of the words generated by the model, and therefore there was no independent way to calculate the optimum, an attempt was made to define them through a Grid Search method (Liashchynskyi & Liashchynskyi, 2019) which we will analyze in detail.

First we will look at parameters defined in respect to the literature and the basic operation of the models. The *vector size* was set to 200, which keeps the dimensional complexity at a low level, but not too low, allowing the model to capture details [22]. The parameter *sg* [23] was set to be present *sg=1* in both models while the workers are standard depending on the computational capabilities of the computer therefore the parameter workers=4. Finally, for the Word2vec model the parameter *seed= 42*, which is a classical (random) setting of this parameter in machine learning (Zhou et al., 2025).

So the hyperparameters that remain to be defined with some basis on which they seem to make the models work optimally are:

1. *window*
2. *min count*
3. *epochs*
4. *negative* (Only for Word2vec)

[22] It is generally recommended to be between 50-200 (Grayson et al., 2016)

[23] The introductory article on the use of Word2vec suggest to use this parameter with small dataset and when we need to manipulate rare words https://code.google.com/archive/p/Word2vec/

Thus, in order to find the optimal distribution of the above three for fastText and four for Word2vec, a Grid Search Method was applied. This method is a classical method for hyperparameter optimization, and what it does, is simply to perform a complete search on a given subset of hyperparameters of the model being trained. But since the parameters in models and ML algorithms can include infinite or non-real numbers, we need to define a search boundary (Liashchynskyi & Liashchynskyi, 2019). We will achieve this result computationally by creating a loop, that compares specific values for each parameter with the others, in order to compare every possible combination of values that we have defined as candidates, to search for each parameter. The values chosen starting from the lowest value to the highest are presented in table 5.8, while the other hyperparameters of the model would remain constant [24]:

Parameter	**Word2vec**	**fastText**
`window`	3,4	3, 4
`min_count`	1,5	1, 5
`epochs`	30, 40	30, 40
`negative`	10, 15	–

5.8.1 Clustering of Embeddings

At this stage, the central question, in this search, is how to define the evaluation of the optimal combination of parameters. So, by seeking to establish a framework for evaluating them in the topic modeling context, that this chapter focuses on, the evaluation of parameters will be inferred from the clusters that each possible combination will produce.

Before we move on to how we clustered the embeddings produced by each model, it is important to give a definition of clustering, and to explain the algorithm we used to perform it here. '*Clustering is the task of partitioning a dataset into groups, called clusters, where data points within a single cluster are more similar to each other than to those in different clusters*'(Müller & Guido, 2016). Clustering and classification are both fundamental tasks in Data Mining. In the chapter 4, where the distinction of Papadiamantis' language was discussed, a classifier model was developed using a supervised learning technique. On the other hand,

[24]The `negative` parameter is applicable only to Word2vec in this comparison.

clustering, similar to the word embeddings approach, is mostly performed through *unsupervised learning* (at least in the present study, this method is adopted). The goal of clustering is descriptive, whereas in classification, it is predictive (Veyssieres & Plant, 1998). Specifically, clustering aims to create new sets of categories, meaning that a specific form of data is grouped into clusters rather than assigned to predefined classes. These new sets, known as clusters, group data in such a way that similar instances belong to the same cluster, while different instances are assigned to separate clusters. Formally, the clustering structure is represented as a set of subsets $C = C_1, ..., C_k$ of S, such that:

$$S = \bigcup_{i=1}^{k} C_i \quad \text{and} \quad C_i \cap C_j = \emptyset \quad \text{for } i \neq j.$$

Consequently, any instance in S belongs to exactly one and only one subset. The central idea of clustering is as ancient as human civilization itself and is strongly related to the inherent need to categorize entities, assigning individuals to the appropriate groups (Rokach & Maimon, 2005). Therefore, with regard to our task, we expect the algorithm to group together, within each cluster, words (and sub-words in the case of fastText) that share semantic similarity, while assigning those that do not to separate clusters.

Let's now take a closer look at how the clustering algorithm we used, called HDBSCAN [25] (Malzer & Baum, 2020), works. First, we load the vector representations produced by each model. Since each representation consists of 200 dimensions, we reduce their dimensionality using one of the following techniques at a time: *PCA* (Jolliffe, 2002; Ringnér, 2008), *t-SNE* (Cai & Ma, 2022), or *UMAP* (McInnes et al., 2018) [26]. Using the reduction tecniques we can keep the semantic relations, but in a manageable number of dimensions in order to make the data '*clusterable*'. Then, the data is ready for clustering by HDBSCAN. Lets keep in our mind, that the algorithm does not have a predefined number of clusters to generate, but by defining *the minimum number of cluster size* and the *Euclidean distance* it is able to generate clusters. This is actually the great advantage of this algorithm, namely that we do not need to predetermine the number of clusters. However, we must define these two parameters. To find the optimal operating condition, we again applied a grid search technique as explained below.

Returning to how the algorithm operates, HDBSCAN first constructs a *mutual-reachability*

[25]The algorithm is available here with instructions of installation and its state-of-the-art: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.HDBSCAN.html

[26]Each method is applied independently to visualize the structure of the embedding.

graph that includes all points. This graph encodes *core distances* (the local density around each point) and *mutual reachability distances* (how reachable two points are from one another based on shared density thresholds). The algorithm then computes a *minimum spanning tree* (MST) over this graph and progressively removes edges, starting with the longest (least dense) connections. As edges are removed, connected components of the graph emerge, representing clusters at different density levels. This process produces a *cluster hierarchy* (or cluster tree dendrogram), from which HDBSCAN selects the most stable clusters, those that persist across a wide range of density thresholds. Points that do not belong to any stable cluster are classified as noise and labeled -1 (Malzer & Baum, 2020).

The main question is how to evaluate clustering output, which utilises an embedding model (Word2vec or fastText), a dimensionality reducer (PCA, t-sne, UMAP) and a clustering algorithm HDBSCAN. Based on this evaluation, we will be able to assert that we are using optimal parameters for our clustering approach, as determined by the grid search across the embedding models, dimensionality reducers, and HDBSCAN [27]. In fact, we need metrics for evaluation and we chose: *Cohesion Score, Silhouette Score and DBCV (Density-Based Clustering Validation Index)*. Thus, we can evaluate each clustering approach and the embedding models in order to obtain the optimal clusters of the present dataset.

Let's see what each metric evaluates exactly starting from *Cohesion score*. Informally, the basic idea behind the mathematics of cohesion stores is that it measures how similar items (here, word vectors) in a cluster are to each other. In essence, it looks at each unique pair of vectors in the clusters and calculates how much these vectors point in the same direction. It then averages these values to get one number. If this number is closer to 1, then the vectors are very similar and the cluster is tight and consistent. Yet, if it is closer to 0 or negative, the vectors are very different and therefore scattered or inconsistent. So, this metric gives us a score to express how internally coherent the cluster is.

Formally, we define the *Cohesion score* of each cluster as follows: Let C_k be a cluster identified by HDBSCAN, containing n_k word vectors $\vec{w}_1, \vec{w}_2, \ldots, \vec{w}_{n_k} \in \mathbb{R}^d$. The intra-cluster cohesion of cluster C_k is defined as the average pairwise *cosine similarity* between all word vectors in the cluster:

$$\text{Cohesion}(C_k) = \frac{2}{n_k(n_k-1)} \sum_{i=1}^{n_k} \sum_{j=i+1}^{n_k} \frac{\vec{w}_i \cdot \vec{w}_j}{\|\vec{w}_i\| \, \|\vec{w}_j\|} \tag{5.1}$$

[27] For the reducers, we use the default parameter values

We see that the two vectors in the equation are multiplied together to compute their dot product, $\vec{w}_i \cdot \vec{w}_j$. The result of this multiplication is then divided by $\|\vec{w}_i\|$, which is the norm (or magnitude) of vector i, and by $\|\vec{w}_j\|$, which is the norm (or magnitude) of vector j. Essentially, the numerator measures how much the two vectors point in the same direction, while the denominator normalizes the result so that the maximum similarity is 1 (when they point exactly the same way). This operation computes the *cosine similarity* between i and j. When the result is 1, the vectors have perfect similarity; 0 means they are orthogonal, i.e., unrelated; and -1 means they point in completely opposite directions.

The double sum:

$$\sum_{i=1}^{n_k} \sum_{j=i+1}^{n_k}$$

ensures that the calculation is performed for every unique pair (i, j) without repetitions or comparing a vector to itself. Finally, the factor:

$$\frac{2}{n_k(n_k - 1)}$$

ensures that the sum is averaged over all pairs in the cluster. Overall, this equation averages the pairwise *cosine similarity* of all items within a cluster. A high *cohesion* value (closer to 1) means the cluster is tight, while a low *cohesion* value (closer to 0 or even negative) means the cluster is more dispersed and scattered.

The overall *cohesion* score across all valid clusters (excluding noise) is then:

$$\text{AvgCohesion} = \frac{1}{|\mathcal{K}|} \sum_{C_k \in \mathcal{K}} \text{Cohesion}(C_k) \tag{5.2}$$

where $\mathcal{K}$ is the set of all non-noise clusters (i.e., clusters with label ≥ 0 and $n_k \geq 2$).

Therefore, with the second equation, i.e. the average of the *cohesion score* of all generated clusters, we can evaluate the performance of each possible combination of parameters from those we seek as optimal. To put it simply, the higher the average *cohesion score*, which ranges from 0 to 1, the more valid semantic associations the members of a cluster have, and can be searched for topics, which is our goal here. However, this metric does not evaluate how well the clusters are separated, but only the intra-clustering similarity.

Regarding the second metric, the *Silhouette score*, this evaluates the geometric quality of

clusters by comparing them internally and externally. It looks at how close one point is to another within the same cluster, e.g. a word to another, but also how close or far away a cluster is from the nearest one, e.g. a set of clustered words from the nearest set of other clustered words (Rousseeuw, 1987). By doing this comparison, it can be measured, in simple words, whether each word has been placed in the appropriate cluster. It ranges from -1 to 1, and the higher the better.

The third and final metric is DBCV (Density-Based Clustering Validation), which evaluates density separation and compactness and is designed specifically for density-based methods such as HDBSCAN or DBSCAN. It evaluates the density of clusters, the density sparseness between clusters, i.e. how well separated the clusters are, and it also handles noise points well (Moulavi et al., 2014). Essentially, this metric evaluates the separation of clusters in terms of data density, ranging from -1.0 to +1.0, with higher values being better.

So, we conducted an experiment to find the best way to create clusters. Like Word2vec and fastText, which require us to find their optimal operating parameters, the same search for optimal operating parameters is needed for the clustering algorithm (HDBSCAN). Therefore, extending the grid search for the embedding models we saw earlier to the clustering algorithm, the summary table is as follows [28]

Parameter	Word2vec / HDBSCAN	fastText / HDBSCAN
window	3, 4	3, 4
min_count	1, 5	1, 5
epochs	30, 40	30, 40
negative	10, 15	–
min_cluster_size	5, 10	5, 10
min_samples	5, 10	5, 10
metric	`euclidean`	`euclidean`
prediction_data	`True`	`True`

Each search for parameter combinations was performed each time with a different dimension reducer. In practice, we evaluated a total of 96 different configurations, 64 for Word2vec and 32 for fastText, by systematically exploring all combinations of the defined hyperparameters.

[28]We used the default parameters for each of the three candidate dimension reducers (PCA, t-sne, UMAP), while training was performed on the first 8,000 words. The dataset consisted of words.

The following diagram visualises the pipeline we followed to find the best combination of word clustering in Papadiamantis' short stories:

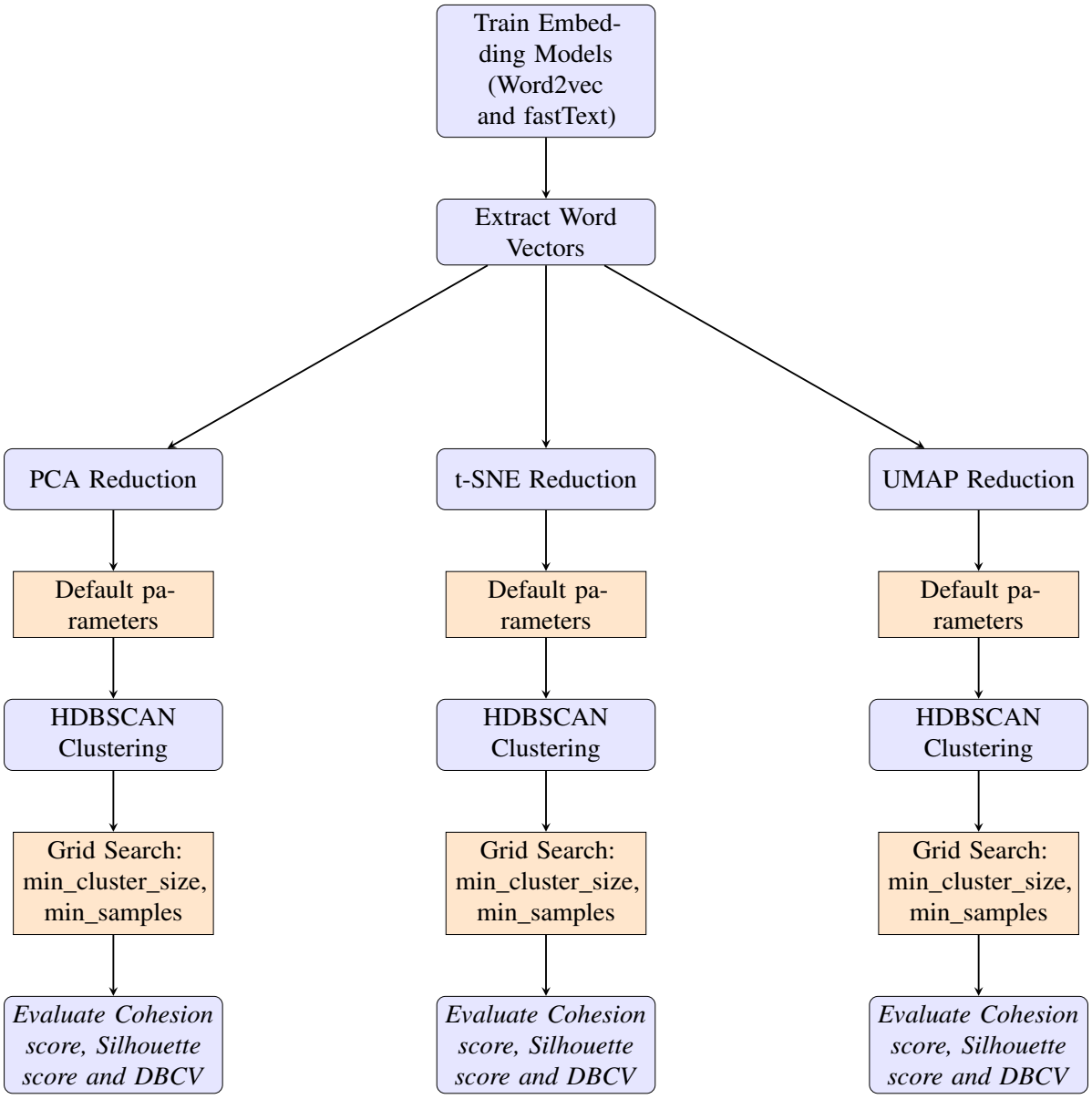


Figure 5.5: Overview of the grid search and clustering process for each dimensionality reduction approach.

5.8.2 Results

After the aforementioned search for the optimal model parameters, we present in the table 5.1 the three best combinations along with their scores in the aforementioned metrics, both for Word2vec and fastText:

Regarding the best combination for Wordvec, we observe that it presents the best *cohesion*

Table 5.1: Top configurations for Word2vec and fastText across different dimensionality reducers. Bold rows indicate the selected best setting for each model.

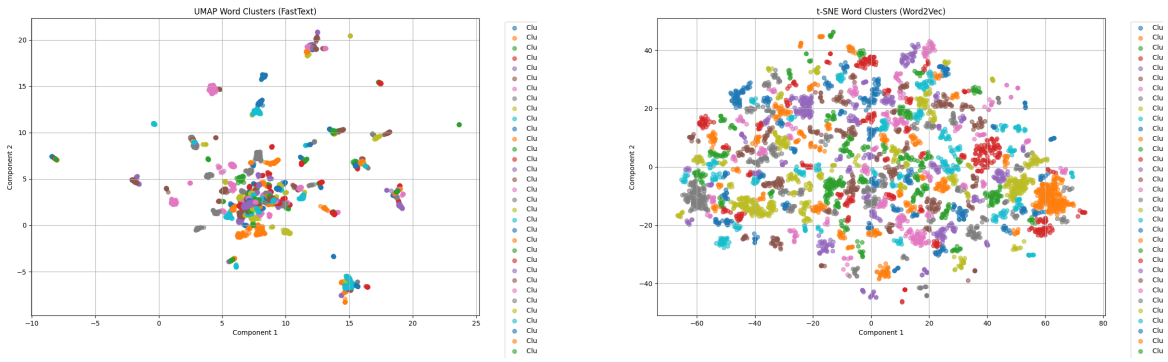
Model	Reducer	Win	MinC	Ep.	Neg	ClustSz	MinSamp	Coh.	Sil.	DBCV	Clusters
Word2vec	**t-SNE**	**3**	**1**	**30**	**10**	**5**	**5**	**0.81**	**-0.19**	**0.15**	**310**
Word2vec	t-SNE	3	5	30	10	5	5	0.79	-0.17	0.18	299
Word2vec	t-SNE	3	1	30	15	5	5	0.79	-0.20	0.18	319
fastText	**UMAP**	**3**	**1**	**30**	**–**	**5**	**5**	**0.62**	**0.46**	**0.55**	**52**
fastText	UMAP	3	1	30	–	10	10	0.59	0.32	0.44	25
fastText	t-SNE	3	1	30	–	5	5	0.68	0.19	0.13	31

score among the rest, a relatively balanced *DBCV*, a rich number of clusters, while the *silhouette score* is relatively negative but acceptable with t-sne. This combination, compared to the others, maximizes the internal stability of the clusters (mainly with *cohesion*, but also with a *silhouette score* that is close to the other two) and the richness of the clusters, with good density-based validation.

On the other hand, regarding fastText, we observe that the optimal combination is lower than that of Word2vec in terms of *cohesion*, but with a higher *silhouette score* and *DBCV*. Compared to the other two combinations, it performs better on all three metrics, while the number of clusters is also richer.

5.9 Topics across Papadiamanti's Short stories

In our interpretative analysis, we will focus on specific clusters from each model. The following diagrams show the clusters produced by each optimal model:



(a) UMAP Word Clusters (fastText)

(b) t-SNE Word Clusters

Figure 5.6: Side-by-side comparison of dimensionality-reduction techniques applied to fastText word embeddings.

In fact, we see that fastText produced fewer clusters than Word2vec. This can be explained

Cluster #	Representative words (Greek + English)
7	έψαλε (sang), ιησού (Jesus), ανέστη (arose), ελέησον (have mercy), δεύτε (come)
43	εορταί (celebrations), κυριακής (Sunday), τυρινής (Christian celebration), φώτων (Lights), απόκρεω (Christian celebration)
73	χήραν (widow), πτωχήν (poor), αδελφήν (sister), μητέρα (mother), γραίαν (old woman)
38	οπτασία (vision), νεύματα (gestures), βλέπη (may see), εφάνη (appeared), όνειρον (dream)
115	αρραβώνας (betrothals), προίκά (dowry), υπανδρεύετο (was married), εορτάση (celebrate), αποθάνη (die)
273	γλάρου (seagull), πελώριος (gigantic), εξετείνετο (extended), άμμος (sand), βράχος (rock)
98	λύπης (sorrow), μανίας (mania), αίσθημα (feeling), ηπείλει (threatened), φόβου (fear)
279	οσφύος (loins), κροτάφους (temples), οφρύς (eyebrow), κόμης (hair), νώτων (backs)
195	νεότητος (youth), ζώσα (living), σκέψις (thought), μητρική (maternal), φαντασίαν (imagination)
172	ορφανών (orphans), αδελφών (siblings) , δελχαρώς (character's name), προστάτης (protector), κόρης (daughter)

Table 5.2: Selected Word2vec clusters with five representative words each.

Cluster #	Representative words (Greek + English)
11	θηρίον (beast), μηναίον (Christian liturgical book), τζαμίον (pane), αρνίον (lamb), ωτίον (ear)
30	κιμωλίασ (chalk), ευκολίασ (ease), θεωρίασ (theory), αγνοίασ (ignorance), θωπείασ (caress)
44	γοητείαν (charm), κηδείαν (funeral), ξυλείαν (timber), ευθυμίαν (cheerfulness), αγγαρείαν (drudgery)
54	εξήρχετο (was going out), διήρχετο (was passing through), εισήρχετο (was entering), επήρχετο (came upon), επανήρχετο (was returning)
178	υγρασία (humidity), αξία (value), κωπία (toil), αφθονία (abundance), σεβασμία (venerable)

Table 5.3: Selected fastText clusters with five representative words each.

by the different function and ultimately different usefulness of fastText compared to Word2vec, showing that it is not as suitable for our interpretative analysis, as will be seen below. With this in mind, from the total number of clusters generated, we selected 10 clusters from Word2vec and 5 from fastText. The tables 5.2 and 5.3 present the clusters from each model, showing 5 representative words from the top words, in terms of semantic representativeness:

Let's start in reverse. Regarding fastText, we observe that the clusters it produces are mainly based on subwords (i.e., morphemes and other sublexical units), and are therefore particularly useful for categorization based on word morphology, but not on their semantic-thematic association. This can be seen in Table 5.3. Specifically, we observe that clusters are consistently categorized according to morphological characteristics. The first cluster (11) is characterized by the ending -ον, the second (30) by the ending -ίας, the third (44) by the ending -ίαν, the fourth (54) by the ending -ήρχετο, and the fifth (178) by the ending -ία. Therefore, for an analysis that attempts to make semantic and broader thematic associations between words, we

consider that this approach, in the present examination of the specific corpus, does not answer our research question, which is to identify distinct thematic directions in the corpus of short stories and to examine the semantic correlation of the words identified in these directions.

In contrast, the Word2vec clusters prove to be significantly more suitable for our analysis, offering clearer semantic groupings that align more closely with our research question as we can see in Table 5.2. First of all, it is important to note that we selected these ten because they interested us, and this is an assumption we can make in computational criticism, namely that we choose to look at aspects of the results that concern us. In our opinion, there are two important issues in these results. The first is that the clustering algorithm, among the set of words, can distinguish and group together words that appear to be thematically related. These can be characterized as topics. The second is that these associations tell us something about the topic itself that the algorithm distinguishes.

Regarding the first issue, we can identify topics such as chanting in Christian liturgy (7), Christian celebrations (43), descriptions of female identity (73), vision and dreams (38), marriage (115), the landscape of the Skiathos (273), negative emotions (98), the description of the female body (279), youth (195) and finally terms related with the short story '*The Murderess*'. Some of these distinct groupings of words are known from the literature to fall under broader topics in Papadiamantis [29].

Let's have a closer look at these topics. First, on the subject of liturgical language (7), we can observe how specific the references to words from hymns in liturgical texts are (ανέστη, ελέησον, δεύτε, ιησού) which are related to the verb έψαλε, showing the real connection between the short stories and the liturgical language of the church and the art of chanting. Christian celebrations (43) are also very specific and refer both to major celebrations throughout the year (Απόκρεω, Φώτων, Τυρινή, as well as more frequent celebrations (Κυριακή, εορταί). We can argue that the time of the stories is greatly determined by Christian celebrations, and this is evident in this particular topic.

The female form is portrayed in topic (78) with age, social, and family characteristics. We can see what Merlier (2005) stated, that we are dealing with types (of women in this case). The

[29]Indicatively: on Papadiamantis' relationship with liturgical chant, among other things: Ζορμπάς (1991) and Μαντάς (1994, 2002), on Christianity in Papadiamantis, among other things: Λορεντζάτος (1994) and Παπαϊωάννου (2005)), on the properties of women, among others Γκασούκα (1995), on the Skiathos landscape in Papadiamantis' work: Παπαδιαμάντης (2005) on vision and dreams, among other references: Χρυσογέλου-Κατσή (2005), on descriptions of the female body in Papadiamantis' short stories, among other references: (Πουρνή, 2024), on the short story '*The Murderess*' among other references: Denik (2014), Politi (2005), Αναγνωστοπούλου (2015), Καρδαρά (2005), and Μιχαλοπούλου (2014)

woman is presented as a widow, poor, a sister, a mother, and an old woman. We do not see any connection between the woman and thematic words related to love, men, romance, elements of nature, or anything similar. Her age, financial situation, the absence of her husband, the presence of her children, or her relationship with other women are always specified. We see that these types, these different representations of women, are grouped together. This particular connection seems to be an evolutionary path towards social (poverty) and family (widowhood) misery in this specific context. The connection does not place women at the center, but rather the roles they may have in Papadiamantis' universe, showing them as dependent on other external factors and not as autonomous individual.

The topic of vision and dreams (38) is particularly interesting. In the above topic, we can argue for the relationship between vision and dreams. The sense of sight in this particular topic becomes a sense of dreamlike vision. The vision that appears to a subject who sees, it becomes a dream (an elusive one at that), while gestures are the only possible means of action in this situation.

Marriage in Papadiamantis's work is a world of commerce. A marriage without a dowry cannot take place and be celebrated appropriately. We can observe this in cluster 115, which deals with the socio-economic, rather than romantic, union of the bride and groom.

Continuing, the Skiathos landscape, as described for instance in the Kastrin short stories, can be observed in the words gathered in cluster 273, where the castle, the area in Skiathos, is described as full of seagulls, huge, rocky, and sandy. What can be said to add this topic to the Papadiamantis analysis is that it shows precisely that the seascape in the stories is depicted from the perspective of someone standing on the coast, rather than from the sea.

Negative emotions in topic 98 are particularly intense. We observe sadness linked to rage and fear as particularly threatening emotions. These intensely charged negative moods are grouped together, revealing a dark side of the mental world of the stories.

Continuing on, in topic 279, we observe that members of the female body are grouped together. We see an objectification of the female body, which is observed very carefully, but not in connection with the character it describes, but on its own. The female body is independent here (in contrast to the hetero-determination of the female form in topic 73), it does not belong to the woman, the mother or the old woman, but is what it is and is interconnected. From a feminist perspective, this particular theme (in combination with 73) could describe the objectification of women.

Next, in topic 195, we see the closeness of the topic of youth to concepts such as liveliness, thought, mother, and imagination. It is worth noting that the adjective '*maternal*' is used here rather than '*mother*', which indicates the close connection of the young person with the mother's actions, rather than with the mother herself. Imagination and vitality are also considered characteristics of youth, showing the close relationship between writing and the preservation of youth.

Finally, in topic 172, we come across words related to the story of the '*murderess*'. Δελχαρώ, the daughter of the murderess in the short story of the same title, is the only mother among her sisters, which sets the scene for the murderess's actions and motivations. The word that particularly struck us in this theme is the word *protector*. Among orphans, siblings, and daughters, a protector appears. Perhaps this topic can support the view that the murderer's actions, in a way that is not apparent to us, had a protective effect on her children, her children's children, and society as a whole.

5.10 Discussion

In this chapter, we performed the task of topic search in Papadiamantis' short stories using embedding models and clustering algorithms. By evaluating three different pipelines using metrics (*cohesion score, silhouette score, DBCV*), the best possible method was used to identify topics in the corpus of Papadiamantis' short stories. We are aware that the result is limited by the use of static embedding models and the reduced number of words after applying the parameters and the stopword list, but we consider that this does not reduce the effectiveness of this topic modeling approach. The combination of Word2vec with the t-SNE dimension reducer and the HDBSCAN clustering algorithm was optimal for our analysis, while fastText was not used for interpretative analysis, despite the fact that it was identified as the optimal method for cluster generation.

We selected ten topics for our analysis, while all identified topic clusters are publicly available to researchers[30]. The contribution of this approach is that by representing the words from Papadiamantis' short stories in a multidimensional (semantic) space, we were able to cluster them in terms of semantic similarity. These clusters seem capable of capturing the topics explored in Papadiamantis' short stories. Moreover, we were able to see the correlations

[30]All topic clusters can be accessed at: https://github.com/dimitrispapad/Papadiamantis/tree/main/Topics%20Identification

between words within these topics and present interpretative suggestions based on this analysis of correlations.

We selected ten clusters that we considered to be topics worth discussing in our literary analysis. We believe that, depending on the research questions and interests of each individual, the same clustering can be viewed differently, and many different interpretations can be made based on it. Through this approach, we did not assume that Papadiamantis' short stories deal with the liturgical language of the church, Christian celebrations, aspects of female identity, dreams and visions, the condition of marriage, the seaside landscape of Skiathos, negative emotions, the female form, youth, and the field of action of *the murderess* based on our own perception of the work, but instead we identified the above in the results of a method evaluated to produce word clusterings.

This analysis focused on these clusters (without using arguments, but only these clusters) to argue that there is a strong connection between Papadiamantis' short stories (1) with the art of chanting and (2) the liturgical life of the church, (3) with the association of women with roles that devalue her, (4) the close relationship between dreams and visions, (5) the socio-economic reality of marriage, (6) the seascape seen from the coast, (7) the dark world of negative emotions, (8) the vitality of youth and its closeness to the mother and the imagination, (9) the objectification of the female body, and finally (10) the protective action of the murderess in the short story of the same title.

Chapter 6

Clustering Papadiamantis' Works with Contextual Embeddings

6.1 Introduction

The categorisation of an author's work is undoubtedly a challenging task within the discipline of Modern Greek literature. In fact, this categorisation must in some way be able to answer the question of whether there are enough distinctive characteristics to categorise something and, if so, on what basis of distinction.

Although it is particularly common for this categorisation to be based either on the chronological correlation of the works or on their themes, here we propose a different method of categorisation through the use of embeddings and clustering of all the works under consideration. This type of categorisation, as will be seen, does not begin with the assumption that categories must be formed at all costs; but after giving a final vector at the end of the process for each of Papadiamantis' works (so that each work is represented by a numerical vector, which can then be examined to see whether distinct categories naturally emerge), it examines whether we can create different categories or not. By using this method, we can both pose and answer the question of whether something has enough distinctive characteristics to be categorised in a broader context in computational literary analysis.

This chapter explores the complex issue of categorising the works of Alexandros Papadiamantis (short stories and novels), approaching the categorisation in a different direction from those already existing in the literature. The final result is the categorisation of the works themselves, as a representation in a multidimensional (semantic space).

6.2 Related Work: Literary Periodization and the Need for Computational Approaches

A particularly controversial issue in Papadiamantis studies is the division of his work. The most extensive categorisation effort sets time periods as the boundaries for distinguishing between works, attempting to identify thematic patterns. Previous attempts to categorise Papadiamantis' works by thematic patterns (e.g., Καστρινά διηγήματα, Αθηναϊκά διηγήματα, Χριστουγεννιάτικα διηγήματα, etc.) group selected stories under specific themes. However, they do not provide a comprehensive categorisation of the author's oeuvre. We therefore exclude them from our literature review, which focuses on holistic approaches to the classification of his works. With regard to the chronological periodization of his work, Στεργιόπουλος (2005) proposes dividing Papadiamantis' work into three phases: The first phase covers the period of the three novels and the short story 'Χρήστος Μηλιώνης' (1879–1885); the second phase includes the short stories from 1887 to 1896; and the third phase spans from 1896 to 1910.

Στεργιόπουλος (2005) states that the first phase includes the three novels ('Οι έμποροι των εθνών', 'Η γυφτοπούλα', 'Η μετανάστις') along with the short story 'Χρήστος Μηλιώνης'. This period is characterised by works that unfold, to a greater or lesser extent, in a historical setting and take place before the Greek Revolution of 1821, offering a picture of Hellenism on land and sea through the stories of its heroes in the prerevolutionary era. The short story 'Χρήστος Μηλιώνης', according to the same scholar, is a landmark work that bridges the gap between the novels and short stories that followed.

The second phase begins with the first short story 'Χριστόψωμο' (1887) and ends with 'Έρως- Ήρως' (1896). Some notable short stories that belong to this phase are 'Υπηρέτρα' (1888), 'Η Σταχομαζώστρα' (1889), 'Μία ψυχή' (1891), 'Φτωχός Άγιος' (1891), 'Η Μαυρομαντηλού' (1891), 'Ο πολιτισμός εις το χωρίον' (1891), 'Θέρος- Έρος' (1891), 'Ο Αμερικάνος' (1891), 'Στο Χριστό στο Κάστρο'(1891), 'Η νοσταλγός' (1892-1894), 'Οι χαλασοχώρηδες' (1892), 'Λαμπριάτικος Ψάλτης' (1893), 'Πατέρα στο σπίτι' (1894), 'Ο έρωτας στα χιόνια' (1895), 'Ο ξεπεσμένος δερβίσης' (1896), 'Το σπιτάκι στο λιβάδι' (1896), "Άγια και πεθαμένα', "Έρως- Ήρως' (1896).

The third phase begins with the short story 'Γουτού-Γουπατού' (1898) and ends with the short story 'Ο αντίκτυπος του νου' (1910), which is also the last short story Papadiamantis wrote while he was ill, days before his death.

According to the same scholar, the following short stories are representative of this phase: *'Τ' αγνάντεμα' (1899), 'Τα δαιμόνια στο ρέμα' (1900), 'Απόλαυσις στη γειτονιά' (1900), "Ονειρο στο κύμα' (1900), 'Κοκκώνα θάλασσα' (1900), 'Η Φαρμακολύτρια' (1900), 'Η τύχη απ την Αμέρικα' (1901), 'Υπό την βασιλικήν δρυν' (1901), 'Το νησί της Ουρανίτσας' (1902), 'Στρίγγλα μάνα' (1902), 'Τα μαύρα κούτσουρα' (1903), 'Η Φόνισσα' (1903), 'Τα κρούσματα' (1903), 'Ο Κακόμης' (1903), 'Η συντέκνισσα' (1903), 'Η Φωνή του δράκου' (1904), 'Γυνή πλέουσα' (1905), 'Ρεμβασμός του δεκαπενταύγουστου' (1906), 'Το μοιρολόγι της φώκιας' (1908), 'Νεκρός ταξιδιώτης' (1910)*

According to Στεργιόπουλος (2005), what distinguishes the second and third phases of Papadiamantis' writing is their movement within the realm of sketch of manners, while at the same time blending sketch of manners features with social and psychographic elements, and alternating between a realistic tone and lyrical extensions. In the first period, Papadiamantis is portrayed as characteristically sketch of manners and more overtly social. In the following two phases, he transcends these boundaries, returning to deeper personal experiences and to Greek reality, while in the final phase his writing takes on a distinctly confessional quality. The comparison between periods also involves the examination of specific themes such as divine providence, evil, sin, the author's unfulfilled eroticism, the tendency for Papadiamantis' characters to belong more to the exception than to the rule, and issues of social injustice.

We consider the topics identified in this study to be particularly useful as suggestions for further research. However, the criteria underlying this categorisation are not clearly defined, the framework has not been tested on a large body of data, and the analysis relies on a selective number of well-known works to sustain its interpretive claims. The proposed division introduces chronological boundaries in Papadiamantis' oeuvre, based on correlations with sketch of manners, social, and psychographic elements. Nevertheless, we argue that such a categorisation would require both qualitative and quantitative investigation, along with precise definitions of what is meant by sketch of manners, psychographic, and social features in the texts, and the extent to which each work presents these elements in measurable ways.

Based on this attempt to analyze the distinction of Papadiamantis' work into phases, but also on the opinions of Ξενόπουλος (2005) and Μουλλας (1974) regarding the lack of fluidity and development in Papadiamantis' work, or as Ξενόπουλος (2005) characteristically states, '*it would be difficult to find in his work any development, movement, or progress of thought and insight*', we believe that this topic is suitable for computational investigation due to the

very different starting points and methods used for categorization in these contexts. Moreover, Piper (2019) also mentions that those moments when a change occurs in an author's corpus are particularly important, as they reveal his openness and point of development, and that this is something we can attempt to investigate computationally.

To the best of our knowledge, no attempt has been made to date to conduct a computational study to classify Papadiamantis' works (short stories and novels). Therefore, this chapter constitutes a first attempt to categorize the works based on semantic similarity and differentiation, as will be explained below, through technical representations of the texts in embeddings and a clustering approach, which will categorize the works.

6.2.1 Computational Framework: From Documents to Clusters

The significance of this task, the clustering of Papadiamantis' novels and short stories in a computational criticism context, lies in the fact that, after representing all the works separately, we can represent each one with embedding models and then apply the clustering algorithm, showing the relationship between the works in the multidimensional semantic space. Through this new perspective of highlighting their interrelationships and representing them, we can interpret both the movement of Papadiamantis' work (grab the metaphor) and the interrelationships between his works.

In fact, we followed an approach similar to that of the previous chapter, but with two key differences. The first is the model used to generate the embeddings, and the second is that the clustering here does not concern words that need to be categorized, but documents, i.e., finite and specific-sized sets of words [1]. Before we look at how clustering was applied, it is particularly important to discuss the model we used, its capabilities for Greek, and its general mode of operation.

6.3 From Static to Contextual Representations

Let us recall from the previous chapter 5 that for Word2vec, the representation of word's meaning is the same vector irrespective of the context. That's also true for fastText, with the difference

[1]The corpus used is the manually collected corpus (see 3.1), as this allows each work to be used as a separate document, unlike the unified CLARIN corpus, with Greeklish names for the works and the date at the beginning. All works have the format: 1908tomyrologitisfokias.txt, while the corpus used includes the 172 short stories and 3 novels.

that it represents subwords instead of words. Now, let us consider those two examples:

(1) a. Το ροδάκινο κόπηκε στα δύο με το μαχαίρι μονομιάς επειδή αυτό ήταν μαλακό.
the peach cut.PASS.3SG in.the two with the knife at.once because it was soft
'The peach was cut in two with the knife at once because it was soft.'

b. Το ροδάκινο κόπηκε στα δύο με το μαχαίρι μονομιάς επειδή αυτό ήταν κοφτερό.
the peach cut.PASS.3SG in.the two with the knife at.once because it was sharp
'The peach was cut in two with the knife at once because it was sharp.'

As we can see, the static vector for words like αυτό *(it)* may somehow encode that this is a pronoun referring to animals or inanimate entities. However, how can we capture a more specific, context-dependent interpretation, as in the examples mentioned above? How can we represent the contextual meaning of each word, as in Example 1.a and 1.b, where in the first case *it* must be associated with the peach, and in the second case with the knife? Another explicit example of the necessity for context-dependent word associations, especially in literary texts, comes from examples such as the following [2]:

(2) a. Η κατάθεση του μάρτυρα ήταν επαρκής
the testimony the.GEN witness.GEN was sufficient
'The testimony of the witness was sufficient.'

b. Η κατάθεση στην τράπεζα ήταν επαρκής
the deposit in.the.ACC bank.ACC was sufficient
'The deposit in the bank was sufficient.'

In both examples, the word ' κατάθεση ' is used, but in each case with a different meaning. In the first, it means the provision of information, mainly sworn, to a public authority (testimony), and in the second with the meaning of the delivery of money by a natural or legal person to a credit institution for safekeeping, interest, and future return (deposit) (Triantafyllides, 1998). However, also in this case, by using static embeddings, we are unable to generate separate vectors for each context-specific semantic differentiation. The point of these examples is that different contexts give words different meaninigs, while these contextual words may be quite far

[2]A similar example in English, as noted in (Jurafsky & Martin, 2024), is the following: *I walked along the pond, and noticed one of the trees along the bank*. In this case, the word *bank* is not associated with any financial institution, but rather refers to the edge of a body of water—once again highlighting the importance of context in determining meaning.

apart in the sentence or paragraph (Jurafsky & Martin, 2024). Therefore, we need something more than static embeddings to capture the context-dependent meaning of words.

6.4 BERT and the Rise of Contextual Language Models

6.4.1 Foundational Concepts Underpinning BERT

The basic models examined and the optimal model used for this type of contextual representation here are multilingual versions of BERT (Devlin et al., 2019). Therefore, in this section, we will examine the state of the art of BERT, within the scope and capabilities of a master's thesis, of course, and not exhaustively, but giving a pretty good idea of how BERT works and the basic mechanisms it uses, so that we can then discuss its multilingual versions, which are used for Greek.

However, in order to discuss how BERT works, we will attempt, within the limits of this thesis, to outline some basic concepts that provide the necessary background for its operation. Striving for a balance between excessive technical detail and oversimplification, we will provide a concise overview of the attention mechanism, transformer-based models, large language models (LLMs), and the principles of pre-training and fine-tuning.

Let's start with the basic concepts. First of all, BERT belongs to the category of transformer-based models. The transformer is the basic architecture for building a large language model (LLM). Essentially, it is a neural network that uses the self-attention or multi-head attention mechanism (we will explain in detail). The basic architecture of a transformer is shown in the figure below, taken from Jurafsky and Martin (2024).

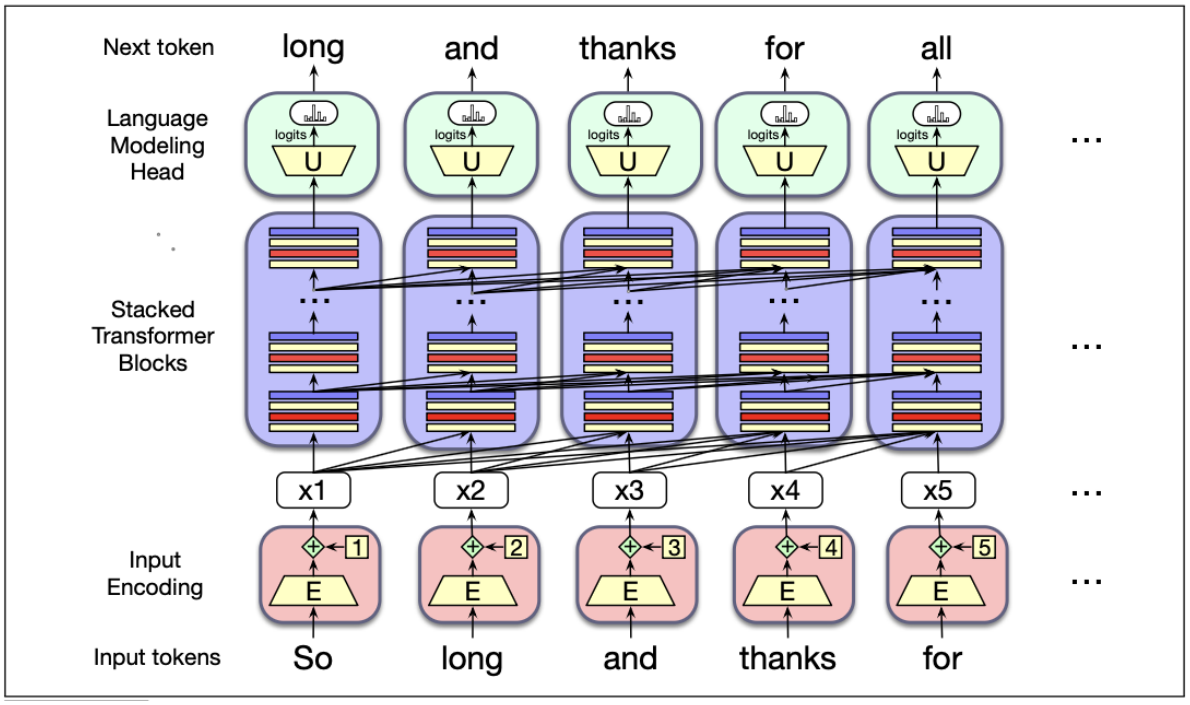


Figure 6.1: The architecture of a transformer model illustrating input encoding, stacked transformer blocks, and the language modeling head (Jurafsky & Martin, 2024).

In the figure above, we see a representation of how a transformer can predict the next token (word in this case) through the three basic components that comprise it. In the center are the columns of the transformer, the blocks. Each block is a multilayer network (multi-head attention layer, feedforward networks [3] along with layer normalization steps [4]) that maps an input vector x_i in column i, which corresponds to input token i, with an output vector h_i. The set of these blocks now forms the entire context window, a group of input vectors and their corresponding output vectors of the same size. Every column can contain from 12 to 96 stacked blocks. These blocks are preceded by the input encoding component, which creates a contextual representation of each token. This is done using an *embedding matrix E*, which gives each word or token a numerical form that the model can understand, and a mechanism that adds information about the position of each token in the sequence. Next, each column is followed by a language modelling head, which takes the outputs from the last block, processes them through an *unembedding matrix U*, and applies a *softmax* function[5].

[3]*'A feedforward network is a multilayer network in which the units are connected with no cycles; the outputs from units in each layer are passed to units in the next higher layer, and no outputs are passed back to lower layers.' (Jurafsky & Martin, 2024)*

[4]By normalization steps, we mean the process of converting a list of numbers into a form that behaves like a probability distribution, that is the values are between 0 and 1 and add up to 1 (Jurafsky & Martin, 2024).

[5]The softmax function turns a list of values, called logits, into a probability distribution. This means that each number is transformed to fall between 0 and 1, and all the values add up to 1. It uses exponentiation, meaning each

Given a vector **z** of length K, the softmax function is defined as:

$$\text{softmax}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{K} \exp(z_j)}$$

over the vocabulary to generate a single token for that column (Jurafsky & Martin, 2024).

Transformers can build contextual representations of word meanings, contextual embeddings, by integrating the meaning of contextual words. In a transformer, layer by layer, contextual representations are enriched by the meaning of input tokens. In fact, in each layer, we calculate the vector representation for a token *i* by combining information about *i* from previous layers with information about neighboring tokens to represent the contextualized representation of each word in each position (Jurafsky & Martin, 2024). The mechanism that plays a central role in the operation of transformers and weights and combines representations from the appropriate other tokens of the preceding context *k-1* to create the representation in layer *k* is called *Attention* (Uszkoreit et al., 2017).

Attention takes an input representation x_i that corresponds to the input token at position *i*, and a context window of the previous inputs $x_i..x_i-_1$, and produces an output α_i. So, when processing an input representation x_i, the model has access to both x_i and the entire preceding context, but not to the tokens following x_i, which, as we will see below, is not the case with BERT. At its core, attention is simply a weighted sum of context vectors, along with many complications added to how the weights are calculated and how we get these summed. By weighing each previous embedding proportionally to how similar or dissimilar it is to the token *i* under consideration, we can see this similarity between the vectors, depending on the size of the dot product, while normalizing these vectors through softmax to obtain the vector of weights. To sum up, the mechanism of *Attention,* in a simplified form, shows that its calculation involves comparing x_i with all previous vectors, normalizing these vectors into a probability distribution using the weighted sum of all previous vectors (Jurafsky & Martin, 2024).

The above is also reflected in the basic version of *Attention* that we encounter in transformers, where an *attention head* includes the *query* (the current element compared to the previous tokens), the *key* (preceding input), and the *value* (of a previous element that is weighted and summed up to compute the output for the current element). Of course, transformers do not use a single *attention head*, but multi-head attention, where each head can focus on a different linguistic

number is treated as an exponent of the constant $e \approx 2.718$, to emphasise the largest values (Jurafsky & Martin, 2024).

relationship between the context elements and the element under consideration, or examine specific types of patterns in the context. In fact, without looking at the calculations in detail, due to the scope of this thesis, what *multi-head attention* does is take a vector x_i as input and map it to an output a_i (attention i) by adding vectors from previous tokens, weighted by how relevant they are to the processing of the specific word (Jurafsky & Martin, 2024).

The multi-head attention layer together with a feedward layer that follows are included in the residual stream of the transformer block in Figure 6.1, where, more broadly, the input from the previous layer passes to the next, with the output of the different elements being added. The transformer blocks are stacked to make the networks deeper and more powerful.

As for the input encoding part in the same figure, this is calculated by adding embedding (which is calculated with the embedding matrix) to the positional encoding, which practically represents the positional location of a token in the window. Finally, language models can build stacks of transformer blocks with a language model head at the top, which applies an unembedding matrix to the output *H* of the top layer to produce logits, which are then passed through softmax to generate probabilities (Jurafsky & Martin, 2024).

To give an idea of how large a window context transformer-based models can process, suffice it to say that this size can start at 200k words, enabling them to process very large volumes of data to make their predictions. BERT is based on this transformer architecture. However, before focusing on BERT, it is useful to discuss two more basic concepts: pre-training and fine-tuning.

The result of pretraining, i.e., the process of learning knowledge about language and the world from a huge amount of text, is LLMs, which can perform a significant number of NLP tasks, such as text generation, language identification, sentiment analysis, etc., without supervision (Jurafsky & Martin, 2024). But how is a transformer trained to be a language model, and what algorithm is used to train it?

In practice, to train a transformer as a language model, we use a corpus as a training tool and at each step *t* we ask the model to predict the next word, without giving the model any gold labels, but using the text itself as supervision. Thus, the model is simply trained to minimize its error in correctly predicting the next word in a sequence using cross-entropy as the loss function (Jurafsky & Martin, 2024). How? Cross-entropy essentially measures the difference between the distribution of prediction probabilities and the correct distribution using the following equation:

$$\text{LCE} = -\sum_{w \in V} y_t[w] \log \hat{y}_t[w]$$

The correct distribution $\hat{y}_t$ results from the correct knowledge of the next word. Thus, if a word is correctly predicted in the vocabulary, the correct vector is 1 and all others are 0. Let's look at an example of how cross-entropy loss works in a language model with the probability assigned by the model to the correct word (Jurafsky & Martin, 2024). For example, if the correct word-class is the word *papadiamantis* and the model has

$$\hat{y}_t[\text{papadiamantis}] = 0.9 \quad \Rightarrow \quad \text{Low Loss}$$

contrary, if it assigns a probability of 0.1, then

$$\hat{y}_t[\text{papadiamantis}] = 0.1 \quad \Rightarrow \quad \text{High Loss}$$

.

The following diagram 6.2 from Jurafsky and Martin (2024) illustrates this training of the transformer as a language model:

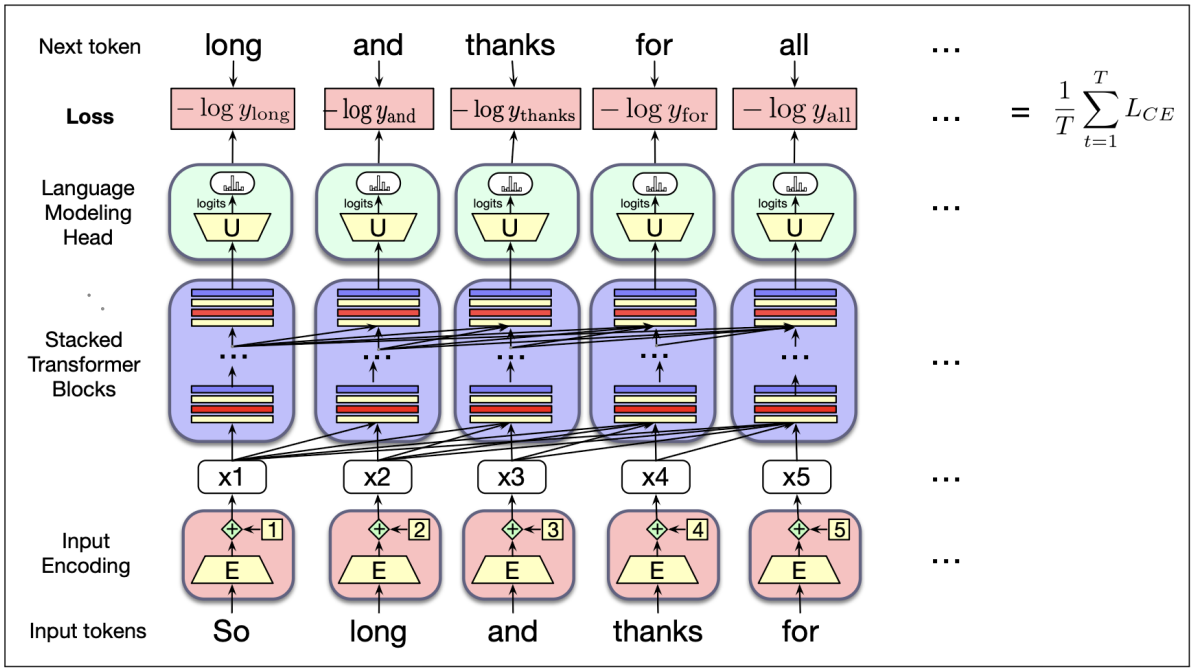


Figure 6.2: Training a transformer as a language model (Jurafsky & Martin, 2024).

However, despite training on a huge amount of data with texts usually from different domains, it is common to apply the trained language model to texts from different domains or to new tasks, which were not satisfactorily present in the training data (Jurafsky & Martin, 2024). For example, we may want to use more texts containing spontaneous speech, perform a next sentence

prediction task, or make our model multilingual. In essence, as Gururangan et al. (2020) argues, in these cases, training continues on new data from a new domain or in a new language, while this process of using a fully pre-trained model that undergoes additional training processes on some new data is called fine-tuning and is illustrated in the following figure:

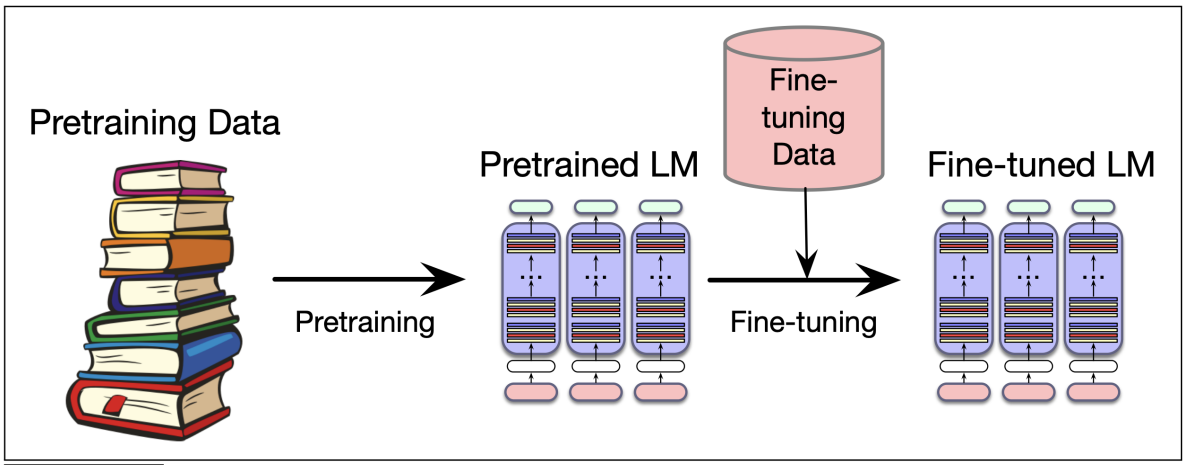


Figure 6.3: Pretraining and finetuning. (Jurafsky & Martin, 2024).

6.4.2 BERT: Bidirectional Encoder Representations from Transformers

In the original paper proposing BERT by Devlin et al. (2019), we can see its contribution to the field of transformer-based LLMs. This particular model is used, in comparison with other LLMs, not to generate text but to understand it. It is based on transformer models but differs initially in that it uses only encoders, and indeed a bidirectional one. The focus of Bidirectional encoders is on computing contextual representations of input tokens, using the self-attention mechanism to map a sequence in the input embeddings to a sequence of the same size in the output embeddings, where the output embeddings have been contextualised using information from the entire input sequence. These output embeddings are contextualised representations. In practice, comparing this to what we saw for embeddings in the previous chapter 5 with their static version, there we represented the meaning of lexical types, while here, in contextual embeddings, we can represent the meaning of the lexical instance, an instance of a specific word or type in a specific context. Therefore, while in Word2vec we had a vector for each instance of this type, here in BERT, the contextual embeddings produced can measure the semantic similarity of words in every context.

By bidirectional, we mean that BERT does not only look at the past context, as we discussed for the architecture of LLMs in the previous subsection 6.4.1. Instead, it has all the input at its disposal, examining the sequence both from left to right and from right to left, while using only

an encoder and not a decoder, since the purpose of basic training is to produce encoding for each token, without producing text as a decoder would do when predicting the next word.

Let's look at the two basic stages of BERT training, namely pre-training and fine-tuning. During BERT pre-training, the model does not predict each subsequent word, but instead chooses to hide 15% of all words in the training set and is trained to predict the missing words, adopting an approach called Masked Language Modelling (MLM), (Devlin et al., 2019; Jurafsky & Martin, 2024). The second task of pre-training is called next sentence prediction (NSP), where after the sentences of the text are randomly divided into those that follow each other (50%) and those that do not follow each other (50%), the model is then asked to predict the next sentenxe.

In both the MLM and the NSP tasks, the loss functions used are variants of the cross-entropy loss. For the MLM task, the loss for each masked token is calculated as:

$$\mathcal{L}_{\text{MLM}} = -\sum_{i \in M} \log P(x_i \mid X_{\setminus M}), \tag{6.1}$$

where M is the set of masked tokens in the input sequence X, and $X_{\setminus M}$ denotes the input sequence with the masked tokens removed.

For the NSP task, the loss is defined as:

$$\mathcal{L}_{\text{NSP}} = -\left[y \log P(\text{IsNext}) + (1-y) \log P(\text{NotNext})\right], \tag{6.2}$$

where $y = 1$ if the second sentence actually follows the first in the original text, and $y = 0$ otherwise. By combining these two losses during pre-training, BERT is trained to effectively capture both token-level and sentence-level dependencies.

During pre-training, BERT was trained on large unlabeled text data. However, in fine-tuning, BERT is trained on additional smaller labelled datasets for specific tasks that evaluate and enhance (as this is a continuation of its training) its ability to understand natural language. Specifically, the training included the GLUE Benchmark, a general collection of datasets for tasks that can be used to test the ability of models to understand natural language, the Stanford Question Answering Dataset SQuAD v1.1, and SQuAD v2.0, i.e. two datasets in which Bert had to find the appropriate answer (in SQuAD v2.0 there are also answers that cannot be answered) and finally Situations With Adversarial Generations (SWAG), where Bert was asked to select the correct answer from multiple answers for the most likely correct continuation of the sentence (Devlin et al., 2019).

Although it is common practice in applications involving text pairs to encode the text pairs independently, before applying bidirectional cross attention, BERT uses the self-attention mechanism to combine these two stages, by combining the text pairs with the self-attention mechanism, to include bidirectional cross attention between the two sentences. For fine-tuning in each task, the appropriate inputs were used depending on the task (sentences A and B from the NSP task of the pretraining) and outputs in BERT, and all the parameters of BERT were fine-tuned end-to-end. The model performance results are available in the original BERT article, while regarding the model size, BERT base includes 110M parameters and BERTLARGE contains 340M parameters (Devlin et al., 2019).

6.4.3 Models Used in Grid Search

While BERT has set the standard for contextual embeddings in English, multilingual variants have been created to address the challenges of low resourse languages like Greek. According to Jurafsky and Martin (2024), multilingual models have an additional choice to make: What data to use in their vocabulary? A common way is to divide the training data into subcorpora of N different languages [6], calculating the number of sentences n_i for each language i and readjusting these probabilities so as to upweight the probability of less-represented languages (Lample & Conneau, 2019).

When training large multilingual language models, we need to feed in text from different languages. However, real-world text is not equally distributed: some languages have lots of data (like English or Spanish) while others might have very little (like Maltese or Icelandic). If we train on this imbalanced data as is, the model will be dominated by high-resource languages and underperform on low-resource languages. To balance this, the training corpus is splitted into N subcorpora, one for each language. Each subcorpus contains n_i sentences from language i. To further adjust the sampling, each language i is assigned a weight parameter a_i that indicates the importance of that language in the training process.

First, a normalized weight p_i is calculated:

$$p_i = \frac{a_i}{\sum_{j=1}^{N} a_j} \tag{6.3}$$

[6] Greek is a low-resource language and is included in multilingual versions of BERT along with other languages.

Then, the probability of selecting a sentence from language i is adjusted as follows:

$$q_i = \frac{p_i n_i}{\sum_{k=1}^{N} n_k} \tag{6.4}$$

This sampling strategy ensures that low-resource languages are upweighted during training, allowing the model to learn better representations for them despite the imbalance in raw data availability.

One might ask why we did not use GreekBERT for this study. GreekBERT[7] is a monolingual model trained exclusively on Greek corpora and has shown strong performance on language understanding tasks such as named entity recognition (NER), part-of-speech tagging, and classification. However, it is primarily designed for token-level tasks and does not natively support sentence-level embeddings, which are essential for clustering tasks based on semantic similarity (Reimers & Gurevych, 2019).

By contrast, the models used in this study, MiniLM (Wang et al., 2020), DistilUSE (Reimers & Gurevych, 2020), and E5-base (Ban & Dong, 2022), are pre-trained to produce sentence-level semantic representations, making them more appropriate for large-scale clustering tasks. MiniLM is lightweight and computationally efficient, ideal for large-scale processing. DistilUSE is specifically designed for capturing sentence-level semantics. E5-base, trained on large-scale retrieval-oriented tasks, provides robust embeddings that capture semantic similarity across languages, including Greek. Thus, our choice reflects the task-specific strengths of these models rather than a limitation of GreekBERT itself.

6.5 Clustering Setup for Papadiamantis' Corpus

In order to perform clustering on Papadiamantis' short stories and novels, we investigated three multilingual embedding models (`DistilUSE-v2`, `MiniLM`, `E5-base`), combined with a single dimensionality reduction method (*UMAP*, 15 dimensions) and two clustering algorithms (*HDBSCAN, k-Means*).

For *HDBSCAN*, we applied the relaxed configuration with `min_cluster_size = 5` and `min_samples = 2`, while for *k-Means* we tested a range of cluster values (`k = 5, 6, 7, 8, 10`). Each configuration was evaluated on the basis of three metrics in the case of *HDBSCAN* (*Silhouette score*, *Davies–Bouldin*, *DBCV*) and two in the case of *k-Means* (*Silhouette score*,

[7] Koutras (2021)

Davies–Bouldin). In addition, both algorithms were systematically assessed with the *Coverage Rate (CR)*, a measure of the proportion of texts successfully assigned to clusters. The tested parameters are summarized in Table 6.1.

Parameter	Sentence Transformers / HDBSCAN	Sentence Transformers / k-Means
`embedding model`	DistilUSE-v2, MiniLM, E5-base	DistilUSE-v2, MiniLM, E5-base
`dim_reducer`	UMAP (15D)	UMAP (15D)
`min_cluster_size`	5	–
`min_samples`	2	–
`metric`	`euclidean`	`euclidean`
`n_clusters`	–	5, 6, 7, 8, 10
`evaluation_metrics`	Silhouette, Davies–Bouldin, DBCV, Coverage Rate	Silhouette, Davies–Bouldin, Coverage Rate

Table 6.1: Parameter settings for clustering with Sentence Transformers, using HDBSCAN and k-Means (with UMAP reduction). Coverage Rate (CR) was added as an additional evaluation metric.

Before moving on to the results, it is important to build on the knowledge gained in the previous chapter and introduce the *k-means* algorithm and the *Davies-Bouldin metric*, since all other elements involved in Grid-search have already been discussed.

6.5.1 k-Means

One of the most fundamental and widely utilized clustering algorithms is *k-means* clustering. In practice, *k-means* seeks to identify cluster centers that best represent distinct regions within the dataset. The algorithm operates through an iterative optimization process, alternating between two key steps: first, each data point is assigned to the nearest cluster center, and then, each cluster center is recalculated by computing the mean position of the data points assigned to it. This iterative process continues until convergence is reached, meaning that the assignment of data points to clusters remains unchanged, ensuring the stability of the clustering structure (Müller & Guido, 2016). The following examples from Müller and Guido (2016) illustrate the application of *k-means* clustering in partitioning data points, showcasing the input data and the two key steps of the *k-means* algorithm:

At the beginning, the algorithm randomly distributes the input data points. Since the number of clusters in this example is set to three, the data points are initially directed toward three centers (the triangles indicate the centers of each cluster, while each color represents the cluster

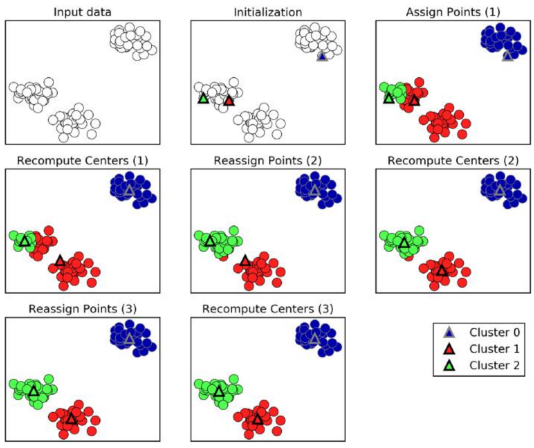


Figure 6.4: Illustration of the clustering process using the k-means algorithm (Müller & Guido, 2016).

membership of each data point). In the '*Assign Points 1*' diagram, the algorithm begins by assigning each data point to its nearest cluster center. Immediately after, the '*Recompute Centers*' diagram shows the step where the cluster centers are updated to reflect the average position of the data points assigned to each one. This process is repeated in the following steps until '*Recompute Centers 3*', at which point the assignment of data points to cluster centers no longer changes, and the algorithm terminates. As a result of this process, we can then define clear boundaries between clusters, as illustrated in the final diagram 6.5 below.

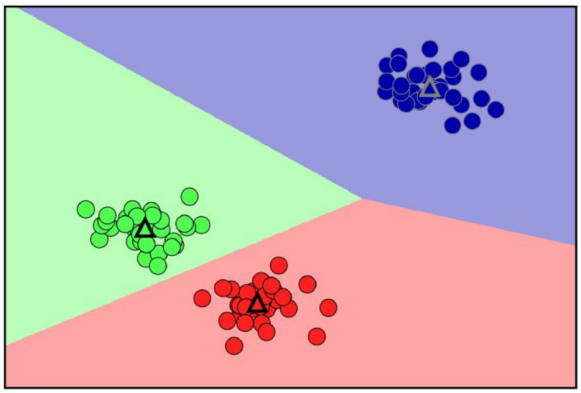


Figure 6.5: Cluster centers and cluster boundaries found by the k-means algorithm (Müller & Guido, 2016).

Finally, an important issue that must be addressed when performing *k-means* clustering is deciding how many clusters should be created, since the number 3 used in the previous example was arbitrary. Ultimately, *k-means* allows for a characterazation of the clusters using the cluster means, or put it simply, we can think about clustering as decomposition method where each data point is represented by its cluster center (Müller & Guido, 2016). Therefore, as in HDBSCAN,

various values were tested for the number of clusters that must be produced, in order for the algorithm to function optimally, in combination with the other preceding stages.

6.5.2 Davies-Bouldin metric

Along with the other two metrics, *Silhouette* and *DBCV*, which we discussed in the previous chapter 5, in this chapter we have added the *Davies-Bouldin metric* (Davies & Bouldin, 2009) as together with *Silhouette*, it can evaluate both clustering approaches, as we can see in 6.1. This metric can evaluate the quality of clustering by providing a more comprehensive picture of the validity of the clusters. Specifically, it evaluates clusters by measuring the average ratio (i.e., the result of the division) of intra-cluster spread to inter-cluster separation. The lower the value, the better, as a low value means that the clusters are compact and well separated. Mathematically, this is expressed as follows:

$$DB = \frac{1}{k} \sum_{i=1}^{k} \max_{j \neq i} \left(\frac{\sigma_i + \sigma_j}{d_{ij}} \right) \tag{6.5}$$

where k is the number of clusters, σ_i is the average distance of points in cluster i to its centroid, and d_{ij} is the distance between cluster centroids i and j (Davies & Bouldin, 2009).

If, for example, we have clusters A, B and C, with cluster A being very tight (the members have a small spread in space) and far away from the other clusters, cluster B having members with a large spread, and being quite close to cluster C, while cluster C is also tight and very close to B, then cluster A will have a low DB index contribution. Cluster B will have a high DB index contribution due to its closeness to cluster C and its large spread. Therefore, by comparing the intra-cluster relationship of the members, as well as the relationship between the clusters, we can finally evaluate the production of the clusters and their quality with this metric.

So, by using these three metrics, we can evaluate the results of the combinations shown in Table 6.1, looking for the best way to cluster Papadiamantis' works.

6.6 Results

Table 6.2 presents the results obtained from the different combinations of clustering configurations, as defined by the chosen parameters of the clustering algorithms, the embedding models, and the dimensionality reduction methods.

Table 6.2: Comparison of clustering configurations across models and methods. Coverage Rate (CR) is included as an additional metric.

Model	Method	Clusters	Noise Points	Silhouette	Davies–Bouldin	DBCV	CR
DistilUSE-v2	**Relaxed HDBSCAN**	**11**	**62**	**0.399**	**0.834**	**0.232**	**0.646**
MiniLM	**Relaxed HDBSCAN**	**3**	**4**	**0.188**	**1.150**	**0.210**	**0.970**
MiniLM	KMeans (k=5)	5	0	0.332	1.097	–	1.000
MiniLM	KMeans (k=6)	6	0	0.321	1.140	–	1.000
MiniLM	KMeans (k=7)	7	0	0.312	1.109	–	1.000
MiniLM	KMeans (k=8)	8	0	0.304	1.136	–	1.000
MiniLM	KMeans (k=10)	10	0	0.281	1.209	–	1.000
DistilUSE-v2	KMeans (k=5)	5	0	0.276	1.279	–	1.000
DistilUSE-v2	KMeans (k=10)	10	0	0.274	1.117	–	1.000
DistilUSE-v2	KMeans (k=8)	8	0	0.265	1.210	–	1.000
DistilUSE-v2	KMeans (k=7)	7	0	0.263	1.201	–	1.000
DistilUSE-v2	KMeans (k=6)	6	0	0.259	1.284	–	1.000
E5-base	KMeans (k=5)	5	0	0.223	1.400	–	1.000
E5-base	KMeans (k=7)	7	0	0.217	1.392	–	1.000
E5-base	KMeans (k=6)	6	0	0.212	1.437	–	1.000
E5-base	KMeans (k=10)	10	0	0.208	1.386	–	1.000
E5-base	KMeans (k=8)	8	0	0.204	1.467	–	1.000
E5-base	Relaxed HDBSCAN	8	69	0.198	1.302	0.097	0.606

In our experiments, we tested multiple embedding models (`DistilUSE-v2`, `MiniLM`, and `E5-base`) and compared their performance using a combination of cluster validity metrics (Silhouette score, Davies–Bouldin index, DBCV) and Coverage Rate (CR). While `DistilUSE-v2` achieved the highest Silhouette scores, its clustering left a substantial proportion of the corpus unclassified (CR ≈ 0.65). By contrast, `MiniLM` produced weaker Silhouette scores but substantially higher coverage (CR ≈ 0.97), ensuring that nearly the entire corpus was included in clusters.

6.7 Applying Clustering in Papadiamantis' Works

Having identified the optimal configuration for clustering, we can now describe in detail how the code processed and separated the 175 text files (172 short stories and 3 novels) into distinct groups.

First, each text is split into overlapping segments through a *chunking* procedure, which ensures that long works exceeding the model's maximum token length can still be represented in full. For each text, these chunks are individually embedded and their vectors are then averaged to produce a single document-level embedding.

Embeddings were generated with multilingual transformer model `MiniLM`), provided through the Sentence-Transformers framework. Each document is thus represented as a high-dimensional

semantic vector that captures sentence-level meaning. To make the clustering computationally feasible and to better reveal the latent structure, dimensionality was reduced to 15 dimensions using *UMAP*.

Clustering was then performed using *HDBSCAN*, with the parameters `min_cluster_size = 5` and `min_samples = 2`. Unlike k-Means, HDBSCAN is density-based and allows for some texts to remain unclustered as *noise*. Given that philological interpretation benefits not only from compact clusters but also from wide corpus coverage, we considered the balance between these two aspects to be decisive. To address the persistent presence of noise points in HDBSCAN, we applied a hybrid approach that combines HDBSCAN with a rescue step using k-Means. The number of rescue clusters was determined dynamically according to the formula:

$$k_{\text{rescue}} = \max\left(2,\ \left\lfloor \sqrt{\tfrac{\text{Noise Points}}{2}} \right\rfloor\right),$$

ensuring that noise points were not arbitrarily lumped together, but rather redistributed into a small number of meaningful clusters proportional to the amount of noise.

This hybrid method (*HDBSCAN + k-Means rescue*) yielded two sets of results: the initial HDBSCAN clusters, and the final rescued clusters including the noise texts. These clusters were then saved both as lists of works per cluster and as interactive two-dimensional visualizations (via UMAP 2D), allowing the relationships among Papadiamantis' works to be explored visually as well as analytically.

6.8 Discussion

The contribution of this chapter lies in exploring a valid way of categorising Papadiamantis' works, along with the categorisation itself. We attempted to achieve the categorisation of the works by using a combination of clustering methods, as was demonstrated in the previous subsection. But what can we now say, interpretively, about this categorisation of Alexandros Papadiamantis' works, in relation to what has been said about their categorisation in the literary criticism?

Initially, despite the fact that Papadiamantis has often been described as an unwieldy and static writer in creative terms, where one might expect similarity across all his works, we can see that they can in fact be divided into fifteen distinct categories in terms of their semantic relationship to each other.

Subsequently, the chronology of the works does not appear to be a factor in our categorisation, since each cluster contains works from different periods of Papadiamantis's writing. The chronological progression of Papadiamantis's oeuvre is therefore not reflected in the present categorisation.

Cluster 0 gathers works that revolve around education, superstition, and the fragility of authority. Tales such as Η Θεοδικία της δασκάλας, Η Δασκαλομάννα, and Τῆς δασκάλας τὰ μάγια situate teachers and pupils in contexts where folklore, satire, and magic undermine the legitimacy of institutions. Alongside them, figures of vulnerability such as the blind poet of Ὁ Τυφλοσύρτης and the threatened children of Ὄχ! Βασανάκια expose the limits of education as moral and social formation. This cluster thus coheres around the schoolroom as a microcosm of wider anxieties, where superstition, satire, and exclusion converge.

Cluster 1 is dominated by social satire and communal disputes, whether through property quarrels (Τὰ Φραγκλέικα, Τὸ κρυφομανδράκι), gossip and naming (Γιὰ τὰ ὀνόματα), or animal symbolism (Γουτουγουπάτου). Here the village emerges as a stage for small rivalries, moral failings, and hypocrisy, often rendered with humor and irony. Figures such as the grotesque Η Καλικατζούνα or corrupt notables in Τ' ἀγγέλιασμα expose the fragility of community bonds when strained by envy, greed, or satire, giving the cluster its distinct thematic profile.

Cluster 2 emphasizes poverty, moral burden, and social despair. Figures of greed and exploitation (Ὁ Γαγάτος καὶ τὸ ἄλογο, Ὁ Πεντάρφανος) coexist with madness and alienation (Ὁ Διδάχος, Ὁ Κοσμολαΐτης). Migration and modern disruption appear in Ὁ Αὐτοκτόνος, while festive contexts such as Ἐξοχικὴ Λαμπρή or Ρεμβασμὸς Δεκαπενταύγουστου underline the tension between ritual continuity and private despair. This cluster coheres through its dark realism, where poverty, despair, and migration intensify the fractures of communal life.

Cluster 3 foregrounds women's lives, marriage, and domestic conflict. Stories like Οἱ Κουκλοπαντρεῖες, Ἡ Ἀποσώστρα, and Ἡ Καλτσὰ τῆς Νοένας expose the burdens of marriage and widowhood, while Οἱ Δύο δράκοι and Θάνατος κόρης dramatize violence and loss. Education, gossip, and female reputation are central to Ἡ Ξομπλιαστῆρα and Ἡ Πιτρόπισσα, highlighting gendered vulnerability. This cluster is unified by its focus on the fragility of women's social position, where domestic life becomes a site of suffering, constraint, and resilience.

Cluster 4 unites stories around faith, ritual, and transgression. Baptism, Easter, and church festivals shape works such as Ἡ Τελευταία Βαφτιστική, Παιδικὴ Πασχαλιά, and Θέρος- Ἔρως,

while disruptive figures haunt Ἡ Φόνισσα, Χωρὶς στεφάνι, or Οἱ Μάγισσες. Satirical depictions of village politics and hypocrisy appear in Οἱ Χαλασοχώρηδες, further complicating the cluster. Its coherence lies in the tension between ritual continuity and disruptive sin or superstition, where faith provides both order and a stage for transgression.

Cluster 5 focuses on wandering, letters, and absence. Works such as Ὁ Ἀειπλάνητος and Ἡ Χολεριασμένη emphasize displacement, sickness, and alienation, while epistolary texts like Τὸ γράμμα στὴν Ἀμερική and miraculous tales such as Τὸ θαῦμα τῆς Καισαριανῆς highlight separation and yearning. Love and death intertwine in Ἡ Ἀγάπη στὸν κρεμνό, producing an atmosphere of rupture and exile. The cluster coheres as a meditation on wandering lives, fractured bonds, and the yearning of letters and miracles.

Cluster 6 is structured by the sea as fate and communal disaster. Shipwrecks, drownings, and storms dominate works like Ναυαγίων-Ναυαγία, Ναυαγοσώσται, and Δημαρχίνα νύφη, while mourning rituals recur in Νεκράνθεμα εἰς τὴν μνήμην των. Figures such as the impious sailor of Ἄψαλτος or the mournful seal of Τὸ Μυρολόγι τῆς φώκιας tie maritime peril to moral and spiritual meaning. The cluster's coherence lies in the depiction of the maritime world as perilous, communal, and spiritually charged.

Cluster 7 dwells on village life, gossip, and kinship. Works such as Ἡ Συντέκνισσα, Ἡ Γλυκοφιλούσα, and Τὰ Πτερόεντα Δῶρα dramatize kinship bonds, gossip, and folklore as formative of community life. Narratives such as Στὸ Χριστὸ στὸ Κάστρο and Τὰ Βενέτικα place these dynamics in larger ritual or historical contexts. The cluster coheres around the representation of village society as a web of rumor, festivity, and kinship.

Cluster 8 turns toward urban encounters and satirical outsiders. Tales such as Ὁ Ἀμερικάνος, Ὁ Πολιτισμὸς εἰς τὸ χωρίον, and Ὁ Ξεπεσμένος Δερβίσης highlight the friction between local life and foreign or marginal figures. Ownership and property disputes shape Τὸ Ἰδιόκτητο, while superstition emerges again in Οἱ Μάγισσες. The coherence rests on the satirical portrayal of outsiders, strangers, and social fault lines that unsettle the boundaries of community life.

Cluster 9 gathers early works of history and nation. In Ἡ Μετανάστις and Ἡ Ἔμποροι τῶν Ἐθνῶν, themes of trade, exile, and national grievance foreshadow later concerns. Ἡ Γυφτοπούλα and Οἱ Παραπονεμένες situate women and communities within proto-national narratives, while satire dominates Ὁ Πανδρολόγος. This cluster holds together as an early exploration of nation, history, and identity.

Cluster 10 emphasizes poverty, eros, and festive ritual. Works like Το σπιτάκι στὸ λιβάδι,

Γιὰ τὴν περηφάνια, and Τὸ Ἐνιαύσιον θῦμα situate mourning and loss within cycles of ritual. Eros under hardship dominates Ἔρως-Ἤρως and Ὁ Ἔρω στὰ χιόνια, while festive joy or light emerges in Φώτα-Ολόφωτα. This cluster's coherence lies in the interplay between love, poverty, and ritual cycles of community life.

Cluster 11 is centered on death, ritual, and the sacred, but also incorporates works of great prominence that situate mortality within the liturgical calendar and communal life. Λαμπριατικὸς ψάλτης, Πάσχα Ῥωμέικο, and Στὴν Ἀγία- Ἀναστασά anchor the cluster in Easter celebrations, where resurrection and communal worship frame life and death in a cyclical ritual pattern. Narratives such as Νεκρὸς Ταξιδιώτης and Τραγούδια τοῦ Θεοῦ focus explicitly on burial, mourning, and psalmody, transforming grief into shared consolation. At the same time, supernatural or uncanny presences recur: Ἁμαρτίας φάντασμα explores haunting guilt, while Ἡ Φαρμακολύτρια stages the destructive powers of sorcery. Domestic hardship is also thematised in Φορτωμένα κόκκαλα, while works like Τὸ Καμίνι and Τ' αερικό στο δένδρο entwine suffering, death, and natural symbolism. This cluster thus coheres around the liturgical and narrative framing of death, Easter, and the sacred, where grief is never isolated but mediated by ritual, memory, and community.

Cluster 12 groups together works that highlight community life, festivity, and satire, often tinged with disorder, conflict, or grotesque exaggeration. Stories such as Ἡ Σταχομαζώστρα and Ο Σημαδιακός portray poverty and fate in village settings, where survival and superstition intersect. Works like Οἱ Χαλασοχώρηδες, Ἐξοχικὸν κρούσμα, and Τὰ καλαμπούρια τοῦ δασκάλου satirize collective life and its hypocrisies, exposing political rivalries, petty quarrels, and social comedy. Festive or liminal moments frame several narratives: Ἀποκριάτικη νύχτα captures carnival disorder, while Ὁ χορός εἰς τοῦ Περίανδρου stages the tensions and humor of a communal gathering. Supernatural or uncanny presences appear in Ἡ Γραῖα κὶ η θύελλα, Τὰ Δύο τέρατα, and Τὸ Ζωντανὸ Κιβούρι μου, where storms, monsters, or the living dead intrude upon everyday life. Finally, Τὰ Τελευταῖα τοῦ Γέρου and Μικρὰ ψυχολογία meditate on aging, death, and decline, placing existential reflection alongside social farce. The cluster thus coheres around the nexus of festivity and community satire, where poverty, conflict, and supernatural disruption intertwine with moments of laughter, ritual, and mortality.

Cluster 13 highlights nature, dream, and mysticism. Ὄνειρο στὸ κύμα, Τὰ δαιμόνια στὸ ρέμα, and Ὑπὸ τὴν βασιλικὴν δρυν interlace natural landscapes with spiritual awe, while Τ' αγνάντεμα extends these concerns into mystic ritual. This cluster coheres around the mystical

and transcendental imagination of nature.

Cluster 14 foregrounds superstition, women, and social strain. Works like Χωρὶς στεφάνι, Ἡ Ἀσπροφοῦστανούσα, and Μάνα καὶ κόρη expose women's burdens within patriarchal society, while supernatural threats dominate Στρίγκλα μάνα and Ἡ Στοιχειωμένη Καμάρα. Festive satire reappears in Τὸ κουκούλωμα. This cluster coheres around the oppression of women through superstition, gossip, and social constraint.

Taken together, these fifteen clusters (0–14) reveal the breadth of Papadiamantis' thematic world. Recurring motifs, ritual, poverty, superstition, women's struggles, migration, and the sea—reappear across clusters, yet each is organized around a distinct center of gravity. Some clusters highlight the maritime and communal dimensions of life (Cluster 6), others foreground women and domestic conflict (Cluster 3, Cluster 14), while others stress festivity and satire (Cluster 12) or mysticism and nature (Cluster 13). The comparative synthesis shows that clustering not only captures thematic regularities but also illuminates the polyphony of Papadiamantis' corpus, where ritual, eros, poverty, exile, and faith intersect in continually reconfigured constellations.

The present close reading interpretation constitutes a first hermeneutic proposal for this categorisation[8]. It does not aim to exhaust the reading nor to provide definitive answers; rather, its purpose is to highlight the interpretive potential that emerges from the clustering of the texts and to stimulate further inquiry. Future work could focus on a more systematic exploration of the common thematic axes and motifs that appear within each cluster, with the aim of mapping more fully the polyphonic world of Papadiamantis.

However, considering the magnitude of this interpretative process, as well as the limitations of a study conducted within the framework of a master's thesis, the contribution of this work remains the categorisation of Papadiamantis' works, not based on thematic motifs or chronological distinctions, but based on a computational method which highlights the semantic relationship between the works and distinguishes them on that basis.

What we have shown is that Papadiamantis' works can be divided into categories based on an optimum model under the given conditions, revealing the semantic relationship between his works when we represent each one as a vector and attempt to categorise them, challenging the argument about the author's static nature and showing, in terms of Piper (2019), his vulnerability

[8]This close reading analysis was assisted by OpenAI's ChatGPT-5 Plus. Specifically, all the examined works were uploaded in groups of ten, accompanied by the following prompt: "The above works constitute the entirety of Alexandros Papadiamantis' prose oeuvre. The categorisation resulting from the optimum clustering algorithm is as follows (each work was provided in a text file together with the cluster to which it belongs). Perform a close reading analysis and identify common thematic associations among the works within the same clusters."

to changement (i.e. sensitivity to change). Chronology does not seem to play a catalytic role in this categorisation. Being aware of the need for an interpretative approach to this categorisation, we offer its results to any researcher who wishes to build interpretative models based on it, having now, however, a method-based modelled approach that can be used for further close-reading analysis.

Chapter 7

Conclusions

The contribution of this thesis is the modeled and computationally based investigation of research questions in Modern Greek literature concerning the work of Alexandros Papadiamantis. In the three main chapters (4, 5, 6), we attempt to answer the following three research questions from the literature: 1) What linguistic variety between Katharevousa, Modern Greek, and Ancient Greek does Papadiamantis use and how does it differ in narrative and dialogue in his short stories and novels? 2) What thematic patterns can be identified in Papadiamantis' short stories? 3) How can his work be categorised? The discussion sections of each chapter contain detailed answers to these questions, so in this chapter we want to place this work in the broader context of Papadiamantis Studies in Modern Greek literature, but also of Computational Criticism more broadly, and to suggest future work.

The present study makes two distinct contributions, both framed under a unified methodological approach. By introducing a language identification task that distinguishes systematically between Katharevousa, Modern Greek, and Ancient Greek in Papadiamantis' prose works, and by applying embeddings, clustering, and grid search both to topic modeling and to the categorisation of his corpus, the thesis demonstrates how computational methods can uncover thematic and structural patterns not easily visible through close reading alone.

First, in the domain of Papadiamantis Studies, this approach provides new evidence on linguistic variety, identifies thematic patterns across the short stories, and highlights corpus-level groupings that challenge strictly chronological divisions. These findings address long-standing debates in Papadiamantis scholarship with computationally grounded insights.

Specifically, with regard to language, the analysis challenges earlier positions in the literature (Άγρας, 1934; Μουλλας, 1974; Ραγκαβής, 1908; Τωμαδάκης, 2005; Ψυχάρης, 1905, among

others) and aligns more closely with interpretations such as those of Tziovas (1989), Πασχάλης (2001), and Τωμαδάκης (2005), which describe the hybridity of Papadiamantis' language and the mixture of oral and textual elements. Nevertheless, this thesis offers a quantified account of the question.

The identified topics and their corresponding interpretive approaches represent a contribution of the present study, strengthening existing close reading interpretations that have discussed similar directions (Denik, 2014; Politi, 2005; Αναγνωστοπούλου, 2015; Γκασούκα, 1995; Ζορμπάς, 1991; Καρδαρά, 2005; Λορεντζάτος, 1994; Μαντάς, 1994, 2002; Μιχαλοπούλου, 2014; Παπαϊωάννου, 2005; Πουρνή, 2024; Χρυσογέλου-Κατσή, 2005, among others).

Finally, the categorisation presented here demonstrates its divergence from chronological classification (Στεργιόπουλος, 2005) and reveals internal differentiation within the corpus itself, in contrast to previously proposed views (Μουλλας, 1974; Ξενόπουλος, 2005).

Second, in the field of Computational Criticism, the study introduces these methods into Modern Greek literature, a largely underexplored area within digital humanities. In particular, it demonstrates how a language identification task can be applied to capture systematic language variation in literary texts, distinguishing between Katharevousa, Modern Greek, and Ancient Greek. More broadly, it shows how embeddings, clustering, and grid search can function as fresh tools for topic discovery and corpus categorisation, where the quality of clusters, assessed through evaluation metrics, strengthens the robustness and interpretive value of the analysis.

With regard to language use, we conclude that the Papadiamantis corpus of short stories and novels contains linguistic elements from Modern Greek, Ancient Greek, and Katharevousa. The dominant variety is Katharevousa. The short stories show a higher proportion of Modern Greek compared to the novels, while in the dialogic passages the use of Modern Greek increases relative to the narrative parts, both in the novels and especially in the short stories. Then, we can also point to a genre-based development of language(and chronological, since the novels precede the short stories), in which the proportion of Modern Greek increases and that of Katharevousa decreases from the novels to the short stories.

With respect to the topics in Papadiamantis' short stories, we provide a detailed account of all identified topics and offer commentary on ten of them. We showed that the short stories are strongly associated with: (1) the art of chanting, (2) the liturgical life of the church, (3) the positioning of women in roles that devalue them, (4) the close connection between dreams and visions, (5) the socio-economic reality of marriage, (6) the seascape viewed from the shore, (7)

the dark world of negative emotions, (8) the vitality of youth and its closeness to the mother and the imagination, (9) the objectification of the female body, and (10) the protective action of the murderess in the short story of the same title.

Regarding the categorisation of Papadiamantis' works, the analysis shows that cluster formation is not driven by chronological proximity or distance of composition. Contrary to views that stress the minimal differentiation and immobility of Papadiamantis' corpus, the analysis reveals that the works can be meaningfully distributed into fifteen distinct clusters. This outcome suggests that, while a broad thematic and stylistic commonality permeates Papadiamantis' writing, sufficiently distinctive patterns nevertheless emerge to allow for coherent categorisation. Each cluster exhibits its own internal cohesion and thematic focus, demonstrating points of differentiation within the corpus. In this sense, the categorisation highlights both the unifying traits that define Papadiamantis' literary voice and the diverse thematic strands that prevent his corpus from being read as static or homogeneous.

We draw inspiration from Moretti (2013), who states that '*we have learned how to read texts up close. We invest our time in years of analysis for a day of synthesis. Let's learn how not to read them.*' Or to put it differently, let's read them in a different way, which is part of the modeled scheme of Piper (2019), combining close and distant reading in a perspective of modeling literature. After all, Moretti (2013) states again, '*Distant reading is a condition of knowledge; it allows you to focus on units that are much smaller or much larger than the text: devices, themes, tropes, genres, systems. Maybe the text itself, or part of it, disappears, but you learn something broader about the system of literature.*'

Given this context, we believe that the present investigation of Papadiamantis' three research questions is based on a new perspective, that of computational criticism, combining both close and distant reading. The language, topics and categorisation of the author's works could not be analysed to such an extent if we did not have these methods of text analysis at our disposal. We know that some kind of simplification, as mentioned in Ramsay (2011), is necessary in text analysis when we attempt to understand something broader about the text or to make generalisations (Piper, 2019). However, as discussed in the introductory chapter 2, the present method is transparent, presenting alongside the interpretation and the observer's position, i.e. the models and datasets that led us to our interpretations.

In the modeled scheme of literary analysis, we consider that we started with a close reading (inspiration for the topic from the bibliography and close reading of the texts), found the

appropriate models to examine our initial research question, performed a distant reading using the models and a large volume of data from Papadiamantis' texts, and arrived at our interpretation.

Some limitations of our analysis are acknowledged and constitute directions for future work. First, the analysis of language use in the corpus was conducted with simple machine learning algorithms, which capture surface-level linguistic features compared to more advanced models (e.g., neural networks). Dialogic passages were separated from narrative ones based on editorial patterns marking dialogue, leaving open the possibility that additional dialogic segments remain within narrative text.

In identifying prominent topics in Papadiamantis' short stories, we selected clusters that are relevant as literary topics. However, the clustering also produced groups of words that may be genuinely connected within the corpus but lack interpretive value. The analysis relied on static embedding models, which cannot capture the contextual variation of word meaning, and was further constrained by a reduced version of the original corpus due to the application of a stopword list and model parameter optimisation.

Finally, the categorisation is limited by the need for closer examination of each cluster individually, in order to ground the grouping in the thematic associations among the works within each cluster. The close reading analysis provided by OpenAI's ChatGPT-5 Plus does not constitute a definitive interpretation, but rather a preliminary ground for exploring the potential of such models in literary analysis. However, its experimental nature, also represents a limitation of this interpretive analysis, as it depends on the mediation of a large language model (LLM) rather than direct human critical engagement. A further limitation arises from the methodological choices adopted. The use of chunking and averaging embeddings across segments, although it allowed us to represent entire works, inevitably smooths over intra-textual variation and may obscure stylistic nuances in longer texts. In addition, the hybrid clustering strategy (HDBSCAN combined with a heuristic k-Means rescue of noise points) ensured full corpus coverage but introduced an element of arbitrariness in the redistribution of noise texts. These trade-offs reflect the balance pursued between semantic precision, interpretability, and completeness in clustering Papadiamantis' corpus.

As future work, we propose a close analysis of the periodisation of Papadiamantis' works, which could be a useful field for close readings, linking this representation of the author's works in the multidimensional semantic space with interpretative analysis. This work could be used to study both the broader evolution of the author's corpus and the individual relationships within

his work. Then, after close reading each cluster, a topic modeling approach can be reapplied to each cluster separately, as in chapter 5, in order to computationally investigate the points of thematic similarity and differentiation between the clusters, answering the central question of what are the actual thematic-literary-motifs that form the basis of this categorization.

Furthermore, we believe that the word groupings could be included in more detailed analyses, as only 10 of them were discussed in this thesis. Finally, it is crucial to address text generation methods, rather than comprehension methods, as was the case with embeddings in the corresponding chapters, and we intend to incorporate Papadiamantis' work into Retrieval-augmented generation (RAG) tasks, where large language models (LLMs) can produce texts in the style of Papadiamantis, identifying and reproducing patterns they find in his texts, and thus identifying both the author's distinctive characteristics and the importance of utilising this approach in the study of modern Greek literature[1].

[1]For more on the first use of RAG in modern Greek literature, see here: Chatzikyriakidis and Natsina (2025).

Bibliography

Abney, S., & Bird, S. (2010). The human language project: Building a universal corpus of the world's languages. *Proceedings of the 48th annual meeting of the association for computational linguistics*, 88–97.

Almeida, F., & Xexéo, G. (2019). Word embeddings: A survey. *arXiv preprint arXiv:1901.09069*.

Angelov, D. (2020). Top2vec: Distributed representations of topics. *arXiv preprint arXiv:2008.09470*.

Ban, Y., & Dong, Y. (2022). Pre-trained adversarial perturbations. *Advances in neural information processing systems*, *35*, 1196–1209.

Bayes, T. (1763). *An essay toward solving a problem in the doctrine of chances* (H. Publishing, Ed.; Vol. 53) [Reprinted in *Facsimiles of Two Papers by Bayes*, 1963]. Hafner Publishing.

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with python: Analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Blacoe, W., & Lapata, M. (2012). A comparison of vector-based representations for semantic composition. *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, 546–556.

Blei, D. (2012). Surveying a suite of algorithms that offer a solution to managing large document archives. cs. princeton. edu 77–84.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, *5*, 135–146.

Browning, R. (1983). *Medieval and modern greek*. Cambridge University Press.

Busa, R. (1980). The annals of humanities computing: The index thomisticus. *Computers and the Humanities*, 83–90.

Cai, T. T., & Ma, R. (2022). Theoretical foundations of t-sne for visualizing high-dimensional clustered data. *Journal of Machine Learning Research*, *23*(301), 1–54.

Chatzikyriakidis, S., & Natsina, A. (2025). Poetry in rags: Modern greek interwar poetry generation using rag and contrastive training. *Proceedings of the 5th International Conference on Natural Language Processing for Digital Humanities*, 257–264.

Chatzikyriakidis, S., Qwaider, C., Kolokousis, I., Koula, C., Papadakis, D., & Sakellariou, E. (2023). Grdd: A dataset for greek dialectal nlp. *arXiv preprint arXiv:2308.00802*.

Chatzivasileiou, V. (2001). Ο Παπαδιαμάντης υπό το βάρος των ερμηνειών του. Η Λέξη *[H Lexi]*, (162), 204–207.

Chatzkyriakidis, S. (2024). *Artificial intelligence and large language models: History, uses, concerns* (M. Deligiannakis, Ed.). Diavlos.

Constantinides, E. (1997). Papadiamantis and the european romantic tradition. *The Journal of Modern Hellenism*, *14*, 1–16.

Davies, D. L., & Bouldin, D. W. (2009). A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2), 224–227.

Denik, I. (2014). Ο *φόνος ως αποστολή: μια κοινωνιολογική και ψυχαναλυτική προσέγγιση του εγκλήματος στην «Φόνισσα» του* Αλέξανδρου Παπαδιαμάντη *και στο «Έγκλημα και Τιμωρία» του* Φιόντορ Ντοστογιέφσκι [Doctoral dissertation, Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης].

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 4171–4186.

Diaz, F., Mitra, B., & Craswell, N. (2016). Query expansion with locally-trained word embeddings. *arXiv preprint arXiv:1605.07891*.

Dimitroulia, T. (2021). Alexandros Papadiamantis: Novels, short stories, poems [Dataset. CLARIN:EL].

Dowty, D. R. (1979). *Word meaning and montague grammar*. Reidel.

Evangelidis, T. E. (1894). Ιστορία του Ιωάννου Καποδιστρίου κυβερνήτου της Ελλάδος*: (1828-1831)* [Accessed: 20.10.2025]. P. E. Zannoudakis Publishing Bookstore. https://anemi.lib.uoc.gr/metadata/3/f/f/metadata-02-0000767.tkl

Févotte, C., & Idier, J. (2011). Algorithms for nonnegative matrix factorization with the β-divergence. *Neural computation*, *23*(9), 2421–2456.

Firth, J. R. (1957a). A synopsis of linguistic theory 1930–1955 [Reprinted in Palmer, F. (ed.), 1968, Selected Papers of J. R. Firth, Longman, Harlow]. In *Studies in linguistic analysis*. Philological Society.

Firth, J. R. (1957b). Papers in linguistics 1934-1951. In F. R. Palmer (Ed.), *Selected papers of j.r. firth, 1952-1959*. Longmans.

Goldberg, Y., & Levy, O. (2014). Word2vec explained: Deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

Grayson, S., Mulvany, M., Wade, K., Meaney, G., & Greene, D. (2016). Novel2vec: Characterising 19th century fiction via word embeddings.

Grootendorst, M. (2022). Bertopic: Neural topic modeling with a class-based tf-idf procedure. *arXiv preprint arXiv:2203.05794*.

Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D., & Smith, N. A. (2020). Don't stop pretraining: Adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*.

Harris, Z. S. (1954). Distributional structure. *WORD*, *10*(2–3), 146–162. https://doi.org/10.1080/00437956.1954.11659520

Hirschberg, J., & Manning, C. D. (2015). Advances in natural language processing. *Science*, *349*(6245), 261–266.

Horrocks, G. (2014). *Greek: A history of the language and its speakers*. John Wiley & Sons.

Jauhiainen, T., Lui, M., Zampieri, M., Baldwin, T., & Lindén, K. (2019). Automatic language identification in texts: A survey. *Journal of Artificial Intelligence Research*, *65*, 675–782.

Jolliffe, I. T. (2002). *Principal component analysis for special types of data*. Springer.

Joos, M. (1950). Description of language design. *The Journal of the Acoustical Society of America*, *22*(6), 701–707.

Jurafsky, D., & Martin, J. H. (2024). *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition with language models* (3rd) [Online manuscript released August 20, 2024]. https://web.stanford.edu/~jurafsky/slp3/

Karkavitsas, A. (1896). *The slender one [*Η λυγερή*]* [Accessed: Accessed: 20.10.2025]. Estia Printing House. https://anemi.lib.uoc.gr/metadata/3/9/4/metadata-141-0000135.tk

Koutras, N. (2021). Greekbert: A greek language model [Accessed: 2025-08-06].

Lample, G., & Conneau, A. (2019). Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Le, Q., & Mikolov, T. (2014). Distributed representations of sentences and documents. *International conference on machine learning*, 1188–1196.

Li, J., Chen, X., Hovy, E., & Jurafsky, D. (2015). Visualizing and understanding neural models in nlp. *arXiv preprint arXiv:1506.01066*.

Liashchynskyi, P., & Liashchynskyi, P. (2019). Grid search, random search, genetic algorithm: A big comparison for nas. *arXiv preprint arXiv:1912.06059*.

Mackridge, P. (1985). *The modern greek language: A descriptive analysis of standard modern greek*. Oxford University Press, USA.

Malzer, C., & Baum, M. (2020). A hybrid approach to hierarchical density-based cluster selection. *2020 IEEE international conference on multisensor fusion and integration for intelligent systems (MFI)*, 223–228.

Marmontel, J. F. (1845). *Belisarius [*Ο Βελισάριος*]* (A. Pangalos, Trans.) [Accessed: 20.10.2025]. Hellenic Commercial School Printing, managed by Alexandros Braun. https://anemi.lib.uoc.gr/metadata/d/5/f/metadata-06-0000062.tk

McInnes, L., Healy, J., & Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Merlier, O. (2005). Από την Σκιάθο το ελληνικό νησί (απόσπασμα) [Απόσπασμα]. In Γ. Φαρίνου-Μαλαματάρη (Ed.), Εισαγωγή στην Πεζογραφία του Παπαδιαμάντη: Επιλογή κριτικών κειμένων *[Introduction to Papadiamantis' Prose: Selected Critical Texts]*. Πανεπιστημιακές Εκδόσεις Κρήτης [University of Crete Press].

Mikolov, T. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, *3781*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, *26*.

Mikros, G. K. (2020). Finding the author of a translation. an experiment in authorship attribution using machine learning methods in original texts and translations of the same author. *Words and Numbers. In Memory of Peter Grzybek (1957–2019)*, 71–82.

Montague, R. (1973). The proper treatment of quantification in ordinary english. In J. M. Jaakko Hintikka & P. Suppes (Eds.), *Approaches to natural language* (pp. 221–242). Springer.

Montague, R. (1974). *Formal philosophy: Selected papers of richard montague* (R. H. Thomason, Ed.). Yale University Press.

Moretti, F. (2013). *Distant reading*. Verso Books.

Moschonas, S. (2016). 'language issues' after the 'language question': On the modern standards of standard modern greek. In *Standard languages and language standards–greek, past and present* (pp. 321–348). Routledge.

Mosteller, F., & Wallace, D. L. (1964). *Inference and disputed authorship: The federalist*. Addison-Wesley Publishing Company, Inc.

Moulavi, D., Jaskowiak, P. A., Campello, R. J., Zimek, A., & Sander, J. (2014). Density-based clustering validation. *Proceedings of the 2014 SIAM international conference on data mining*, 839–847.

Müller, A. C., & Guido, S. (2016). *Introduction to machine learning with python: A guide for data scientists*. " O'Reilly Media, Inc.".

Osgood, C. (1957). The measurement of meaning. *Urban /Universit of Illinois*.

Papadiamantis, A. (1981–1988). Άπαντα *[Complete Works]: Critical edition by N. D. Triantafyllopoulos* (N. D. Triantafyllopoulos, Ed.; Vols. 5) [Critical edition, 5 volumes, in Greek]. Domos.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543.

Piper, A. (2015). Novel devotions: Conversional reading, computational modeling, and the modern novel. *New Literary History*, *46*(1), 63–98.

Piper, A. (2019). *Enumerations: Data and literary study*. University of Chicago Press.

Politi, G. (2005). Δαρβινικό κείμενο και η «Φόνισσα»: Darwinian text and Papadiamantis' *The Murderess*. In Γ. Φαρίνου-Μαλαματάρη (Ed.), Εισαγωγή στην Πεζογραφία του Παπαδιαμάντη: Επιλογή κριτικών κειμένων *[Introduction to Papadiamantis' Prose: Selected Critical Texts]* (pp. 155–181). Πανεπιστημιακές Εκδόσεις Κρήτης[University of Crete Press].

Ramsay, S. (2011). *Reading machines: Toward and algorithmic criticism*. University of Illinois Press.

Reimers, N., & Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*.

Reimers, N., & Gurevych, I. (2020). Making monolingual sentence embeddings multilingual using knowledge distillation. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Ringnér, M. (2008). What is principal component analysis? *Nature biotechnology*, *26*(3), 303–304.

Rokach, L., & Maimon, O. (2005). Clustering methods. *Data mining and knowledge discovery handbook*, 321–352.

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, *20*, 53–65.

Saeed, J. I. (2015). *Semantics* (Vol. 25). John Wiley & Sons.

Schmidt, B. (2015). *Word embeddings*. Retrieved January 14, 2025, from https://bookworm.benschmidt.org/posts/2015-10-25-Word-Embeddings.html

Schnabel, T., Labutov, I., Mimno, D., & Joachims, T. (2015). Evaluation methods for unsupervised word embeddings. *Proceedings of the 2015 conference on empirical methods in natural language processing*, 298–307.

Schreibman, S., Siemens, R., & Unsworth, J. (2008). *A companion to digital humanities*. John Wiley & Sons.

Shakespeare, W. (1889). Hamlet: Tragedy / shakespeare, verse translation by iakovos polylas, with introductions and critical notes [Αμλέτος: τραγωδία / Σαικσπείρου, έμμετρος μετάφρασις Ιακώβου Πολυλά, με προλεγόμενα και κριτικάς σημειώσεις] [Accessed: 20.10.2025]. https://anemi.lib.uoc.gr/metadata/e/f/5/metadata-438-0000067.tkl

Triantafyllides, G. (1998). *Dictionary of standard modern greek*. Institute for Modern Greek Studies of the Aristotle University of Thessaloniki.

Trikoupis, S. (1860-1862). *History of the greek revolution [*Ιστορία της Ελληνικής Επαναστάσεως*]* [Accessed: 20.10.2025]. Taylor; Francis at the Red Lion Court Press. https://anemi.lib.uoc.gr/metadata/2/c/f/metadata-01-0000761.tkl

Turian, J., Ratinov, L., & Bengio, Y. (2010). Word representations: A simple and general method for semi-supervised learning. *Proceedings of the 48th annual meeting of the association for computational linguistics*, 384–394.

Tziovas, D. (1989). Residual orality and belated textuality in greek literature and culture. *Journal of Modern Greek Studies (JMGS)*, *7*(2).

Uszkoreit, J., et al. (2017). Transformer: A novel neural network architecture for language understanding. *Google AI Blog*, *31*, 2017.

Van Rijsbergen, C. J. (1975). *Information retrieval*. Butterworths.

Veyssieres, M., & Plant, R. E. (1998). Identification of vegetation state and transition domains in california's hardwood rangelands. *University of California*, *101*.

Vlachos, A. (Ed.). (1901). *Anthology: Social scenes and studies [*Ανάλεκτα*:* Κοινωνικαί εικόνες και μελέται*]* (Vol. 1). P.D. Sakellarious Press.

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., & Zhou, M. (2020). Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*.

Zhou, H., Savova, G., & Wang, L. (2025). Assessing the macro and micro effects of random seeds on fine-tuning large language models. *arXiv preprint arXiv:2503.07329*.

Άγρας, Τ. (1934). Αλέξανδρος Παπαδιαμάντης. In Θρησκευτική και Ηθική Εγκυκλοπαίδεια *[Religious and Moral Encyclopedia]* (Vol. 9). Συλλογικό Έργο.

Αναγνωστοπούλου, Ι. Κ. (2015). Ύπαρξη και χώρος στον Αλέξανδρο Παπαδιαμάντη *(*Η Φόνισσα*,* η Γυφτοπούλα*)* [Doctoral dissertation, Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης].

Βαλέτας, Γ. (1941). Ο άνθρωπος και η εποχή του [the man and his time]. Νέα Εστία *[Nea Estia]*, *30*.

Βιζυηνός, Γ. (1883). Ποίος ήτον ο φονεύς του αδελφού μου [Serialized publication. Accessed on: [20.10.2025]]. Εστία *[Estia]*, (408). https://pleias.library.upatras.gr/index.php/estia/article/view/69636/61985

Βλαστός, Π. (1911). Για τους κριτικούς του Παπαδιαμάντη [On Papadiamantis' Critics] [30.1.1911]. Νουμάς *[Noumas]*, *9*.

Βυζάντιος, Δ. Κ. (1876). Βαβυλωνία *[babylonia]* (11th edition) [Available in digital format. Digitized by the University of Crete Library]. Εκ του Τυπογραφείου Π. Β. Μωραϊτίνη. https://anemi.lib.uoc.gr/metadata/1/4/1/metadata-265-0000311.tkl

Γκασούκα, Μ. (1995). Η κοινωνική θέση των γυναικών στο έργο του Α. Παπαδιαμάντη [Doctoral dissertation, ΕΚΠΑ – Σχολή Επιστημών Αγωγής – ΤΠΔΕ].

Δάφνης, Σ. (2005). Το χιούμορ του Παπαδιαμάντη. In Γ. Φαρίνου-Μαλαματάρη (Ed.), Εισαγωγή στην πεζογραφία του παπαδιαμάντη*:* Επιλογή κριτικών κειμένων *[introduction to pa-*

padiamantis' prose: Selected critical texts] (pp. 147–155). Πανεπιστημιακές Εκδόσεις Κρήτης [University of Crete Press].

Εγγονόπουλος, Ν. (1999). Σημειώσεις *[notes]*. Ίκαρος.

Ελύτης, Ο. (1997). Η μαγεία του Παπαδιαμάντη *[The Magic of Papadiamantis]*. Ερμείας.

Ζορμπάς, Β. Α. (1991). Η γλώσσα της Αγίας Γραφής και των λειτουργικών βιβλίων στο έργο του Παπαδιαμάντη [Doctoral dissertation, Πανεπιστήμιο Αθηνών – Τμήμα Φιλολογίας].

Καμπατζά, Β. (2011). Αλέξανδρος Παπαδιαμάντης: Η ανίχνευση του κωμικού στοιχείου στο έργο του. Σοκόλης.

Καρδαρά, Α. (2005). Η «Φόνισσα» του Αλέξανδρου Παπαδιαμάντη: συσχετισμός της με πραγματικές ψυχο-εγκληματικές, γυναικείες μορφές. Σύγχρονη Εκπαίδευση: Τρίμηνη Επιθεώρηση Εκπαιδευτικών Θεμάτων, (141), 120–134.

Κοραής, Α. (1964). Άπαντα τα πρωτότυπα έργα *(τ. α1-α2):* Εθνεγερτικά και πατριωτικά δοκίμια, πολιτικο-κοινωνικοί διάλογοι, αυτοσχέδιοι στοχασμοί για την παιδεία και γλώσσα, ανθρωπιστικά προλεγόμενα, αντιρρητικά και απολογητικά κατά κοδρικά φυλλάδια, μύθοι, στιχουργήματα (Γ. Μ. Βαλέτας, Ed.) [Αναστύλωσε και έκρινε Γ. Βαλέτας]. Δωρικός.

Λορεντζάτος, Ζ. (1994). Αλέξανδρος Παπαδιαμάντης Β' *[Alexandros Papadiamantis II]*. Δόμος.

Μαντάς, Ά. Γ. (1994). Ο «Αυτοδίδακτος» και «Ιδιορρυθμος» ψάλτης Αλ. Παπαδιαμάντης [Reprint from Nea Estia, issue 1564, April 1994]. Νέα Εστία *[Nea Estia]*, (1564), 589–597.

Μαντάς, Ά. Γ. (2002). Η «φωνή» του ψάλτη. Εφημέριος.

Μιχαλοπούλου, Σ. Χ. (2014). Το πρόβλημα του κακού στο έργο του Αλέξανδρου Παπαδιαμάντη [Doctoral dissertation, Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης].

Μουλλας, Π. (1974). Α. Παπαδιαμάντης, αυτοβιογραφούμενος (Vol. 29). Ερμής.

Νάκας, Θ. (2003). Μελετήματα για τη γλώσσα και τη λογοτεχνία. In Γλωσσοφιλολογικά δ' (pp. 441–600). Νάκας Θανάσης.

Ξενόπουλος, Γ. (2005). Το έργον του Παπαδιαμάντη. In Γ. Φαρίνου-Μαλαματάρη (Ed.), Εισαγωγή στην Πεζογραφία του Παπαδιαμάντη: Επιλογή κριτικών κειμένων *[Introduction to Papadiamantis' Prose: Selected Critical Texts]*. Πανεπιστημιακές Εκδόσεις Κρήτης [University of Crete Press].

Παπαγιώργης, Κ. (1997). Αλέξανδρος Αδαμαντίου Εμμανουήλ (2η). Εκδόσεις Καστανιώτη.

Παπαδιαμάντης, Α. (2005). Τα καστρινά: Διηγήματα. Αναπτυξιακή Σκιάθου.

Παπαδιαμάντης, Αλέξανδρος. (2005). Χρήστος Μηλιώνης. Πελεκάνος. (Original work published 1885)

Παπαϊωάννου, Μ. Μ. (2005). Η θρησκευτικότητα στον Παπαδιαμάντη [Religiosity in Papadiamantis]. In Γ. Φαρίνου-Μαλαματάρη (Ed.), Εισαγωγή στην Πεζογραφία του Παπαδιαμάντη: Επιλογή κριτικών κειμένων *[Introduction to Papadiamantis' Prose: Selected Critical Texts]*. Πανεπιστημιακές Εκδόσεις Κρήτης [University of Crete Press].

Πασχάλης, Σ. (2001). Αλέξανδρος Παπαδιαμάντης: Σκοτεινά Παραμύθια, Εισαγωγή, επιλογή, σχόλια *[Alexandros Papadiamantis: Dark Tales, Introduction, Selection, Comments]*. Μεταίχμιο.

Πουρνή, Γ. Ε. (2024). Η εν οίκω ελευθερία – ρωγμές ελευθερίας των νεαρών εφήβων ηρωίδων στα σκιαθίτικα διηγήματα του Παπαδιαμάντη. ΚΕΙΜΕΝΑ για την έρευνα, τη θεωρία, την κριτική και τη διδακτική της Παιδικής και Εφηβικής Λογοτεχνίας, 20–38.

Ραγκαβής, Κ. (1908). Νέα ζωή [αλεξάνδρειας]. Νέα Ζωή*[Nea Zwi]*, *4*(44).

Στεργιόπουλος, Κ. (2005). Ο Παπαδιαμάντης σήμερα – Διαίρεση και χαρακτηριστικά της πεζογραφίας του. In Γ. Φαρίνου-Μαλαματάρη (Ed.), Εισαγωγή στην Πεζογραφία του Παπαδιαμάντη: Επιλογή κριτικών κειμένων *[Introduction to Papadiamantis' Prose: Selected Critical Texts]*. Πανεπιστημιακές Εκδόσεις Κρήτης [University of Crete Press].

Τερζάκης, Ά. (1937). Το πρόβλημα Παπαδιαμάντη [The Papadiamantis Problem]. Νεοελληνικά Γράμματα *[Neohellenic Letters]*.

Τριανταφυλλόπουλος, Ν. (2011). Ο Παπαδιαμάντης του Ζήσιμου Λορεντζάτου. Ίκαρος.

Τωμαδάκης, Ν. Β. (2005). Αλέξανδρος Παπαδιαμάντης [Όπως αναφέρεται στο λήμμα Αλέξανδρος Παπαδιαμάντης, Θρησκευτική και Ηθική Εγκυκλοπαίδεια, τ. 9, σσ. 1166–1188]. In Γ. Φαρίνου-Μαλαματάρη (Ed.), Εισαγωγή στην Πεζογραφία του Παπαδιαμάντη: Επιλογή κριτικών κειμένων *[Introduction to Papadiamantis' Prose: Selected Critical Texts]* (pp. 155–181). Πανεπιστημιακές Εκδόσεις Κρήτης [University of Crete Press].

Φαρίνου-Μαλαματάρη, Γ. (2005). Εισαγωγή στην Πεζογραφία του Παπαδιαμάντη: Επιλογή κριτικών κειμένων *[Introduction to Papadiamantis' Prose: Selected Critical Texts]*. Πανεπιστημιακές Εκδόσεις Κρήτης [University of Crete Press].

Χελιδώνη, Σ. Σ. (2010). Μια εκκλησιαστική παράδοση κρυμμένη στο «Κοινωνικόν Μυθιστόρημα» του Αλέξανδρου Παπαδιαμάντη Η Φόνισσα. Κονδυλοφόρος*[Kondyloforos]*, *9*, 179–180.

Χρυσογέλου-Κατσή, Ά. (2005). Γ. Δροσίνης – Α. Παπαδιαμάντης: Οπτασίες και όνειρα. Παρουσία. Επιστημονικό περιοδικό του Συλλόγου Διδακτικού Προσωπικού Φιλοσοφικής Σχολής Πανεπιστημίου Αθηνών, 495–504.

Ψυχάρης, Γ. (1905). Ρόδα και Μήλα. Νουμάς *[Noumas]*.