UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

BOOK INDEXING, MARGINALIA AND QUESTION/ANSWERING WITH LARGE LANGUAGE MODELS

Diploma Thesis

Sevasti Dimopoulou

Supervisor: Manolis Vavalis

July 2025

UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

BOOK INDEXING, MARGINALIA AND QUESTION/ANSWERING WITH LARGE LANGUAGE MODELS

Diploma Thesis

Sevasti Dimopoulou

Supervisor: Manolis Vavalis

July 2025

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΕΥΡΕΤΗΡΙΑΣΗ, ΠΛΑΓΙΟΤΙΤΛΟΙ ΚΑΙ ΕΡΩΤΗΣΕΙΣ/ΑΠΑΝΤΗΣΕΙΣ ΒΙΒΛΙΩΝ ΜΕ ΜΕΓΑΛΑ ΓΛΩΣΣΙΚΑ ΜΟΝΤΕΛΑ

Διπλωματική Εργασία

Σεβαστή Δημοπούλου

Επιβλέπων: Μανόλης Βάβαλης

Ιούλιος 2025

Approved by the Examination Committee:

Supervisor **Manolis Vavalis**
Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Elias Houstis**
Professor Emeritus, Department of Electrical and Computer Engineering, University of Thessaly

Member **Georgios Thanos**
Laboratory Teaching Personnel, Department of Electrical and Computer Engineering, University of Thessaly

Acknowledgements

I am sincerely grateful to Professor Manolis Vavalis and the esteemed committee members for their invaluable guidance and support throughout the course of this thesis.

I am also deeply thankful to my family for their unwavering encouragement and support, which have been essential to the completion of this work.

DISCLAIMER ON ACADEMIC ETHICS AND INTELLECTUAL PROPERTY RIGHTS

«Being fully aware of the implications of copyright laws, I expressly state that this diploma thesis, as well as the electronic files and source codes developed or modified in the course of this thesis, are solely the product of my personal work and do not infringe any rights of intellectual property, personality and personal data of third parties, do not contain work / contributions of third parties for which the permission of the authors / beneficiaries is required and are not a product of partial or complete plagiarism, while the sources used are limited to the bibliographic references only and meet the rules of scientific citing. The points where I have used ideas, text, files and / or sources of other authors are clearly mentioned in the text with the appropriate citation and the relevant complete reference is included in the bibliographic references section. I also declare that the results of the work have not been used to obtain another degree. I fully, individually and personally undertake all legal and administrative consequences that may arise in the event that it is proven, in the course of time, that this thesis or part of it does not belong to me because it is a product of plagiarism».

The declarant

Sevasti Dimopoulou

Diploma Thesis

BOOK INDEXING, MARGINALIA AND QUESTION/ANSWERING WITH LARGE LANGUAGE MODELS

Sevasti Dimopoulou

Abstract

This thesis examines the effectiveness of Artificial Intelligence (AI) in general and Generative Artificial Intelligence in particular on automated generation of book indexing, book marginalia, related Questions and their Answers and other supporting services.

We design, implement a prototype system, evaluate its capabilities and elucidate unclear issues. The philosophy of our study is that our system's role is to aid rather than replace human expertise. We aim therefore that the synergy between our system and human expertise creates a harmonious balance. It will allow us to utilise the efficiency and automation of associated software tools while capitalising on the cognitive skills and contextual understanding unique to human experts and professionals.

We will investigate the benefits that statistical analysis, established AI tools, existing and emerging Generative AI models and beyond, will lead us to added value book services of superior quality and accuracy.

Keywords:

Generative AI, Large Language Models, Information Systems, Scientific Publishing

Διπλωματική Εργασία

ΕΥΡΕΤΗΡΙΑΣΗ, ΠΛΑΓΙΟΤΙΤΛΟΙ ΚΑΙ ΕΡΩΤΗΣΕΙΣ/ΑΠΑΝΤΗΣΕΙΣ ΒΙΒΛΙΩΝ ΜΕ ΜΕΓΑΛΑ ΓΛΩΣΣΙΚΑ ΜΟΝΤΕΛΑ

Σεβαστή Δημοπούλου

Περίληψη

Η παρούσα διατριβή εξετάζει την αποτελεσματικότητα της Τεχνητής Νοημοσύνης (ΤΝ) γενικά και της Παραγωγικής Τεχνητής Νοημοσύνης ειδικότερα στην αυτοματοποιημένη παραγωγή ευρετηρίων βιβλίων, περιθωρίων βιβλίων, σχετικών ερωτήσεων και των απαντήσεών τους και άλλων υποστηρικτικών υπηρεσιών.

Σχεδιάζουμε, υλοποιούμε ένα πρωτότυπο πληροφοριακό σύστημα, αξιολογούμε τις δυνατότητές του και διευκρινίζουμε ασαφή ζητήματα. Στόχος της μελέτης μας είναι ότι ο ρόλος του συστήματός μας είναι να βοηθήσει και όχι να αντικαταστήσει την ανθρώπινη εμπειρογνωμοσύνη. Επομένως, στοχεύουμε ώστε η συνέργεια μεταξύ του συστήματός μας και της ανθρώπινης εμπειρογνωμοσύνης να δημιουργήσει μια αρμονική ισορροπία. Θα μας επιτρέπει να αξιοποιήσουμε την ταχύτητα και την αυτοματοποίηση του λογισμικού, ενώ παράλληλα να αξιοποιείσουμε τις γνωστικές δεξιότητες και την κατανόηση του πλαισίου που είναι μοναδικές στους ανθρώπινους εμπειρογνώμονες και επαγγελματίες.

Θα διερευνήσουμε τα οφέλη που προσφέρει η στατιστική ανάλυση, τα καθιερωμένα εργαλεία τεχνητής νοημοσύνης, τα υπάρχοντα και αναδυόμενα μοντέλα Γενετικής Τεχνητής Νοημοσύνης και πέραν αυτών που θα μας οδηγήσουν σε υπηρεσίες βιβλίων προστιθέμενης αξίας ανώτερης ποιότητας και ακρίβειας.

Λέξεις-κλειδιά:

Γενετική Τεχνητή Νοημοσύνη, Μεγάλα γλωσσικά μοντέλα, Πληροφοριακά συστήματα, Επιστημονικές εκδόσεις

Table of contents

Acknowledgements **ix**

Abstract **xii**

Περίληψη **xiii**

Table of contents **xv**

List of figures **xix**

List of tables **xxi**

Abbreviations **xxiii**

1 Introduction **1**

1.1 Thesis Objective 2

1.1.1 Contribution 3

1.2 Thesis Structure 3

2 Concepts, AI background needed and objective's importance **5**

2.1 Introduction 5

2.2 Basic Concepts and Technologies 6

2.2.1 Artificial Intelligence 6

2.2.2 Machine Learning 6

2.2.3 Neural Networks 7

2.2.4 Deep Learning 7

2.2.5 Natural language processing (NLP) 7

2.2.6 Generative AI 8

2.2.7 Generative AI and LLMs 8
2.2.8 Definition and Significance of LLMs 9
2.2.9 Basic Characteristics of LLMs 10
2.2.10 Existing LLMs and Their Characteristics 13
2.3 Indexing 14
2.4 Summarization 15
2.5 Marginalia 16
2.6 Questions & Answers 16

3 State of the Art 17
3.1 Introduction 17
3.2 Book Indexing Technology 17
3.2.1 Automated Indexing 18
3.2.2 Automated Indexing - Current Technological Approaches 18
3.2.3 Tools for Book Indexing 19
3.2.4 Recent Research Approaches of Automated Book Indexing 21
3.2.5 Creation of Marginalia Using LLMs 23
3.3 LLMs in Question Answering Systems 25
3.3.1 LLMs Capacities in QA 25
3.3.2 Limits and Challenges 25
3.3.3 Typical Issues 26
3.3.4 Improvement Prospects 26
3.4 Conclusion 26

4 Specifications and User Needs 29
4.1 Introduction 29
4.2 User Groups and Needs 29
4.2.1 Authors and Publishers 29
4.2.2 Students and Educators 30
4.3 System Specifications 30
4.4 Limitations and Assumptions 31
4.5 Use Cases 31
4.6 Conclusions 31

5 Implementations **33**

5.1 System Implementation 33

5.2 Data Source and Text Processing 33

5.3 Key Concept Indexing Subsystem 34

5.4 Marginalia Generation Subsystem 36

5.5 Subsystem for Generating Section Summaries 38

5.6 Subsystem for Generating Section-Based Question–Answer Pairs 39

5.7 Implementation Conclusions 41

6 Evaluation **43**

6.1 Introduction 43

6.2 Evaluation of the Automated Index 43

6.2.1 Evaluation Methodology 43

6.2.2 Evaluation Results 44

6.2.3 Visualization of Evaluation Results 44

6.2.4 Comment on Page Numbers 46

6.2.5 Evaluation of Computational Performance 46

6.3 Evaluation of Generated Summaries and Marginalia 47

6.3.1 Results 47

6.3.2 Scope of Evaluation and Computational Assessment 49

6.4 Evaluation of Generated Section Summaries 49

6.5 Evaluation of Generated Question–Answer Pairs 51

6.5.1 Computational Performance 53

7 Conclusions **55**

7.1 Summary and Conclusions 55

7.2 Future Work 56

Bibliography **59**

APPENDICES **65**

A Source Code and Outputs Access **67**

B Book Indexing – Creating a Semantic Index with LLM **69**

C Marginalia – Generating Paragraph Titles with LLM **73**

D Question and Answer Generation from Summarized Book Sections **77**

List of figures

2.1 Position of Large Language Models (LLMs) within the broader AI taxonomy. Image courtesy of the authors (Shahab et al., 2024, ResearchGate). 9

2.2 The encoder-decoder structure of the Transformer architecture. Image courtesy of Vaswani et al., 2017 (Attention Is All You Need). 11

2.3 Categories and characteristics of language modeling approaches, ranging from statistical models to large language models (LLMs), along with their main applications. Image courtesy of *A Survey on Large Language Models: Applications, Challenges, Limitations and Practical Usage*. 14

6.1 Graphical Representation of the Automated Index Evaluation 45

6.2 BERTScore (F1) per generated title for the first 51 text chunks 48

6.3 The BERTScore (F1) for the summaries of the 20 sections 50

6.4 BERTScore (F1) per question - comparison with summary 52

6.5 Q&A generation success rate (left) and answer accuracy (right) 52

B.1 Two-terminal setup during semantic chunk processing: on the left, Ollama server with API calls; on the right, the indexing script parsing and processing each paragraph. 71

C.1 Generated heading and summary for a paragraph chunk. 74

C.2 Real-time heading generation using Ollama (left: server log, right: chunk processing). 75

D.1 Sample output from `qna_from_summaries.txt` showing two questions and answers per section. 78

D.2 Phase 1 – Section-level summarization using `summary_per_section.py`. 79

D.3 Phase 2 – Q&A generation based on summaries using `generate_qna.py`. 79

List of tables

6.1 Evaluation results of the automated index 44
6.2 Average Evaluation Scores for Summaries and Titles 47
6.3 Average Evaluation Scores for Section Summaries 50
6.4 Summary of Q&A Evaluation Results 51

Abbreviations

i.e.	that is
e.g.	for example
AI	Artificial Intelligence
CNNs	Convolutional Neural Networks
DL	Deep Learning
IR	Information Retrieval
KEA	Keyphrase Extraction Algorithm
LLMs	Large Language Models
MAUI	Multi-Purpose Automatic Topic Indexing
ML	Machine Learning
NNs	Neural Networks
NLP	Natural Language Processing
QA	Question Answering
Q&A	Questions and Answers
RNNs	Recurrent Neural Networks
TF-IDF	Term Frequency-Inverse Document Frequency
API	Application Programming Interface
BERT	Bidirectional Encoder Representations from Transformers
BERTScore	Bidirectional Encoder Representations from Transformers Score
CPU	Central Processing Unit
GPU	Graphics Processing Unit
GPT	Generative Pre-trained Transformer
OCR	Optical Character Recognition
ROUGE	Recall-Oriented Understudy for Gisting Evaluation
Chunking	Dividing text into manageable parts

Prompting	Providing input instructions to guide LLM output
Summarization	Condensing long texts into shorter forms

Chapter 1

Introduction

Nowadays, information organization and retrieval is one of the most important fields of Information Science and Artificial Intelligence. The volume of available data keeps increasing rapidly, particularly in the area of digital books and libraries. Therefore, there is an increasing need for effective mechanisms of book indexing and other tools which are useful for books, such as systems that create marginalia and question/answering (QA) based on text. The traditional methods of classification and retrieval are based on human intervention, rendering the process time consuming, not applicable to large data sets and usually subjective.

In the context of this thesis, the term marginalia refers to short descriptive titles, also known as marginal headings or marginal titles, which appear alongside paragraphs or sections to summarize their content and assist with document structure and navigation.

Given the recent technological advancements, automated book indexing aims to create thematic indices and keywords that allow users to find easily and rapidly the information they are looking for. Marginalia and QA are also aiming to facilitate the process for the user. The methods that are being used vary, from verbal approaches, based on predefined lexicons, to advanced techniques of machine learning and deep learning which take into consideration the semantics of the texts.

Although various technologies have been developed in this field, tasks such as automated book indexing, the generation of marginalia and question/answer creation continue to face considerable challenges. The polysemy and synonymy of terms, the need to adapt to different thematic fields, the understanding of hierarchy of senses and the incorporation of context are some of the basic problems that need to be resolved. The present thesis utilizes contemporary techniques with the aim of improving indexing, generating marginalia automatically

and producing question/answer pairs, while also contributing to the automation of these processes. The challenge that we are facing is the fact that the texts used in the study involve legal terminology and are written in Greek.

1.1 Thesis Objective

The objective of the present thesis is the development of an automated system for book indexing, production of marginalia and creation of QA using Large Language Models (LLMs). The issues that we attempt to resolve are the following:

- **Time consuming and manual process of book indexing**: the creation of analytic book indexing, marginalia and question/answering requires considerable time and human labor, especially for long, extensive books or text collections.
- **Lack of standardized tools of automation**: Despite the rapid progress of Natural Language Processing (NLP) techniques, there is yet no widespread, practical solution that offers a totally automated process of book indexing and production of marginalia and Questions and Answers (Q&A) applying to Greek texts.
- **Preservation of semantic coherence and quality**: The automatically produced information has to preserve a high level of relevance and accuracy, in order to be used in academic and professional settings/environments.

In order to resolve these issues, we propose a methodology that is based on the use of LLMs which are adapted with appropriate programming prompting so they produce:

- Comprehensive summaries per paragraph
- Marginalia that imprint the main topic of each paragraph
- Questions and answers that cover essential points of the text and can be utilized for educational purposes or review

The thesis includes the development of a system that:

- Reads and process PDFs of Greek books,
- Divides the text into smaller units (paragraphs, sections),

- Creates automatically marginalia and Q&A for each individual unit,
- Evaluates the coherence and quality of the derived results using comparison metrics.

Therefore, the thesis aims to contribute to the field of Library Science, educational technology and methods of automated text processing via innovative utilization of LLMs.

1.1.1 Contribution

The contribution of the thesis can be summarized as follows:

1. The existing literature on methods of automated book indexing and production of marginalia and Q&As using Large Language Models was studied.
2. A processing system of Greek books in PDF format with automated division of the text into functional units was developed.
3. Pipelines prompting were designed and implemented to:
    - Extract key concepts for indexing,
    - Create marginalia,
    - Create Q&A.
4. Prompting methods were applied and adapted in order to achieve accuracy and relevance of the derived results for Greek texts.
5. The quality of the derived results was evaluated via use of automated evaluation of text metrics.
6. A model system was created that can serve as a basis and can be improved in order to apply easily to new books with minimal human intervention, contributing therefore to the wider automation of the book indexing process and creation of educational materials.

1.2 Thesis Structure

This thesis is structured in a way that facilitates the reader's understanding of the research process and its results. Each chapter is designed to gradually guide the reader from

the necessary theoretical foundation to the implementation and, ultimately, the evaluation of the proposed solution. More specifically:

Chapter 2: Concepts, AI background needed and objective's importance

This chapter presents the fundamental concepts related to artificial intelligence and large language models that are used in the development of the proposed system. It also explains the significance of such a system and the problems it aims to address.

Chapter 3: State-of-the-Art

This chapter provides an overview of the existing literature and relevant research efforts. Its objective is to highlight the methods and techniques proposed to date, as well as to identify the research gap that this thesis seeks to address.

Chapter 4: Specifications and User Needs

This chapter defines the system requirements based on the user needs. It presents the main user groups and describes the core functionalities that the application must support.

Chapter 5: Implementations

This chapter focuses on the technical description of the implementation. It analyzes the individual methods used for the development of each subsystem, the tools employed, as well as the process of integrating LLMs for the generation of summaries, titles and question–answer pairs.

Chapter 6: Evaluation

This chapter presents the evaluation methodology and results. It includes an analysis of the system performance based on objective metrics, in order to assess the effectiveness and precision of the generated output.

Chapter 7: Conclusions

The final chapter summarizes the research findings and highlights the contribution of this work. It also outlines the limitations of the implementation and proposes potential directions for future research and system improvement.

Chapter 2

Concepts, AI background needed and objective's importance

2.1 Introduction

The rapid development of *Artificial Intelligence (AI)* has influenced considerably the way we interact and organize information nowadays. Thanks to this development, we now have *Large Language Models*, which have emerged as a pioneering and innovative tool in various fields, such as Natural Language Processing (NLP), Questions and Answers systems and Information Retrieval (IR). Utilizing enormous volumes of data and sophisticated architectural neurons, these models become capable of understanding and producing language that resembles human language [1]. This capacity makes them useful and even necessary tools in various fields which up to now were using more traditional methods, such as book indexing. LLMs play a decisive role bridging semantic terms in addition to the better comprehension of context. Their capacity to create dynamic and relevant connections with the context in texts has introduced new approaches to indexing, surpassing the limitations of older systems. Furthermore, they allow us to create fast marginal titles or in general marginalia assisting in this way to a more efficient organization of knowledge. Their contribution is also significant to the systems of Question Answering (QA), given that they are able to extract information from large bodies of texts in real time. They process the text fast and they are able to create questions and answer them. Therefore, they constitute very good tools which offer users more and easier access to knowledge.

In order to better understand the contribution of *LLMs* to the field of indexing, it is nec-

essary to present a theoretical background which covers the basic concepts of *Artificial Intelligence*, *Machine Learning*, *Neural Networks*, *Deep Learning* and *Generative AI*.

The present chapter provides a concise and comprehensive description of these concepts, focusing on *LLMs*, analyzing their significance to the applications that relate to the topic of the current thesis.

2.2 Basic Concepts and Technologies

2.2.1 Artificial Intelligence

Artificial Intelligence (AI) is defined as the science and mechanism that creates 'smart' machines which are capable of performing actions that require human intelligence, such as decision making, recognition of patterns and comprehension of natural language [2]. AI consists of subareas such as Machine Learning, Neural Networks, Deep Learning and Natural Language Processing, etc.

2.2.2 Machine Learning

Machine Learning (ML) is a subfield of Artificial Intelligence that focuses on the creation of algorithms and systems that 'learn' using data. In order to achieve this, a sufficient amount of examples is provided so that the system can be trained.

The major approaches to Machine Learning include:

- **Supervised Learning**: Training using labeled data and their corresponding desired outcomes
- **Unsupervised Learning**: Training using unlabeled data with the goal of identifying hidden structures and patterns. In this case, the model does not have information regarding the outcomes.
- **Reinforcement Learning**: Training of a model that interacts with the environment, receiving rewards for correct decisions [3].

2.2.3 Neural Networks

Neural Networks (NNs) are computational models influenced by the operations of the human brain. They consist of neurons that are organized in layers and are connected with each other via weights. Neural Networks are fundamental to Deep Learning, allowing the modeling of complex relations in data [3].

Major Types of Neural Networks

Multilayer Perceptrons (MLPs) **MLPs**: Multilayer Perceptrons are the most basic type of neural networks. They consist of multilayered neurons fully connected to each other. They are mostly used for indexing and regression tasks. However, their lack of specific structures cause limitations, especially when we work with sequential or grid-like data, such as images.

Recurrent Neural Networks (RNNs) **RNNs**: Recurrent Neural Networks are appropriate for sequential data, such as texts or time series. In this type of networks, each neuron takes information not only from the current input but also from the output of the prior elements within the sequence and as result they allow the network to 'remember' prior elements of sequence [4].

Convolutional Neural Networks (CNNs) **CNNs** are ideal for grid-like data, such as images and videos. Instead of connecting fully each neuron with the following one, they use convolutions to detect local characteristics, such as edges and patterns. In this way, they exhibit a better performance in identifying images and objects [5].

2.2.4 Deep Learning

The term **Deep Learning (DL)** refers to a subfield of Machine Learning which uses neural networks with multiple layers. This approach allows models to learn from big and complex data and understand complex patterns and relationships [5]. Most of the modern advancements of Artificial Intelligence, including LLMs, are nowadays supported by DL.

2.2.5 Natural language processing (NLP)

NLP is a field of AI that focuses on the interaction of computers and human language, as well as on the capacity of computational systems to understand, interpret and produce

human language in a way that is useful and significant for the implementation of a plethora of applications, such as machine translation, sentiment analysis and speech recognition [6].

Basic techniques of NLP

Syntactic Analysis The analysis of syntax examines the structure of sentences and words for grammatical correctness [7].

Semantic Analysis Semantic analysis focuses on semantic comprehension of words and sentences, as well as its applications to natural languages [6].

2.2.6 Generative AI

The term **Generative AI** refers to algorithms that create new content, such as text, images, sound and codes, based on patterns of data training [8]. One example is ChatGPT which can generate human language answers, whereas DALL-E generates images based on descriptions.

The basic characteristics of Generative AI include:

- **Self-supervised Learning:** Models are trained on huge databases to comprehend and produce content.
- **Output Variation:** Generative AI is able to produce multiple versions of content from one input, giving the impression of creativity [8].
- **Applications:** Generative AI is used in fields such as content creation, code development and optimization of business processes.

2.2.7 Generative AI and LLMs

Large Language Models, such as GPT, is a type of Generative AI that focuses on the comprehension and production of natural language. LLMs are using the architecture Transformers and are trained in huge databases, developing capacities such as:

- **Pre-training and Fine-tuning:** Training in enormous volume of data and adaptation for specific tasks [9].
- **Zero-shot Learning:** Performance of new tasks without further training.

- **Creativity:** Capacity of high-quality content creation which resembles human writing.

These specific characteristics will be further analyzed below.

Before we proceed with the analysis of the characteristics of LLMs, it's useful to visualize their position within the broader ecosystem of Artificial Intelligence. The diagram below presents the hierarchical relationship between the core fields of AI, with an emphasis on the place of Large Language Models (LLMs):

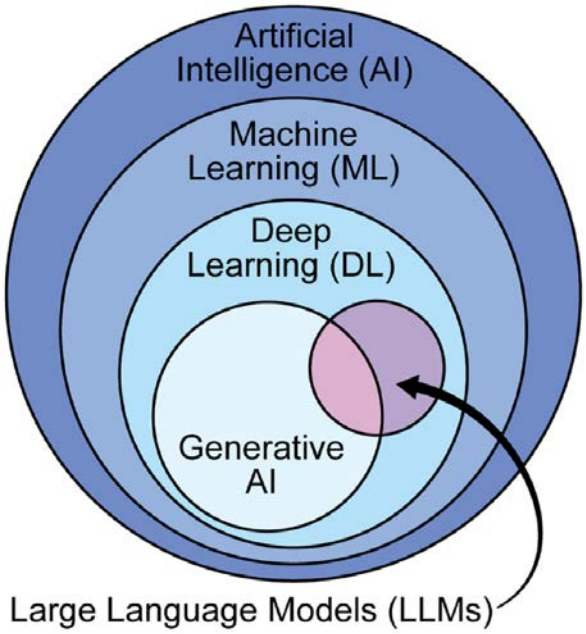

Figure 2.1: Position of Large Language Models (LLMs) within the broader AI taxonomy. Image courtesy of the authors (Shahab et al., 2024, ResearchGate).

As illustrated, LLMs form a subset of Generative AI systems, which in turn fall under the area of Deep Learning. Deep Learning is part of the broader field of Machine Learning, which itself is a subfield of Artificial Intelligence.

2.2.8 Definition and Significance of LLMs

What does the term Large Language Models refer to?
LLMs are types of NNs that are designed to process and produce natural language text. These models are trained in huge numbers of texts and include hundreds of billions of parameters. Their training is taking place in various stages, with the major goal to obtain a general representation of language that can be applied on a large number of NLP tasks, such as text creation, indexing, question answering and summary.

Their significant effectiveness is due to their ability to perform exceptionally on various-language tasks, without having specialized training for each specific task (zero-shot learning). As such, they constitute especially powerful tools for many applications in the field of Computer Science, research and technologies such as artificial intelligence and machine learning [10].

2.2.9 Basic Characteristics of LLMs

The basic characteristics of LLMs that will be described concern the architectures being used for their implementation, pre-training and fine-tuning. All of the above are crucial for the performance and the capacities of these models.

Transformer Architecture

The architecture of Transformers is a revolutionary architecture which was introduced in 2017 by the article [11]. Thanks to this architecture, the traditional RNNs and CNNs used for a number of applications have been largely replaced. The architecture is based on the separation of two main blocks: the **Encoder** which encodes the input text and the **Decoder** which is responsible for producing the final output. In what follows, we will analyze the basic components.

Tokenization and Embeddings

Let us examine what happens when the input text is inserted into the Transformer. The input is a sequence of text which initially is broken down into **tokens**, i.e. into smaller units, such as words or subwords. For each token, a high dimensional representation will be created, called **embedding**. Embeddings are mathematical vectors responsible for encoding the semantic information of the token [12]. Transformers use contextual embeddings, that is dynamic representations which adapt depending on the frame of use.

Encoder: Encoding the Input Text

The **Encoder** transforms the embeddings in abstract representations which yield the information of the input text. An important aspect of the Encoder's function is the mechanism of **Self-Attention**. Thanks to this mechanism, the model can recognize relations between words

in the same sentence. For instance, words, such as *bank*, can be analyzed differently depending on the frame [13]. Additionally, Position Embeddings are imported to encode the order of tokens, given that the Transformers cannot understand where each token is in a sequence.

Decoder: Creation of Final Output

The **Decoder** receives the abstract representations from the Encoder and then creates the final output, such as a translated sentence, or an answer to a question. The Decoder uses the mechanism of **Cross-Attention** which combines information from the embeddings of the input and the output that has already been created. In this way, the creation of a coherent and relevant sequence is ensured [14].

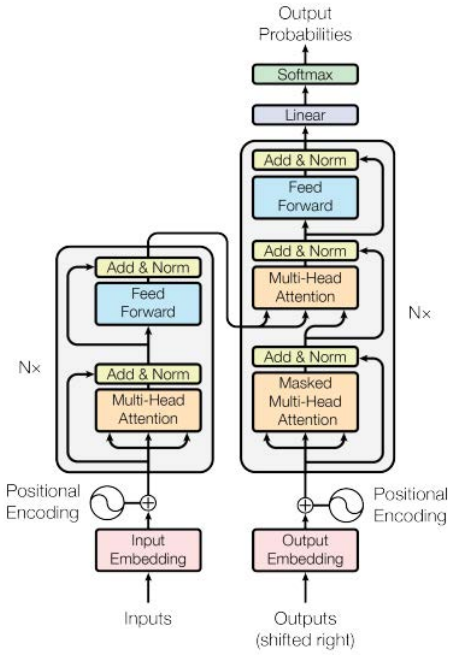

Figure 2.2: The encoder-decoder structure of the Transformer architecture. Image courtesy of Vaswani et al., 2017 (Attention Is All You Need).

Attention Mechanism: The Core of the Architecture

The **Attention Mechanism** is the foundation of the architecture of Transformers. Via **Self-Attention**, the Encoder estimates the relations of tokens in a sentence, whereas the Decoder uses **Cross-Attention** to connect the inputs and the outputs. The innovation of **Multi-Head Attention** lies in the capacity of the model to analyze multiple relations simultaneously, improving therefore the comprehension of complex sentences [11].

Position Embeddings

Transformers cannot process data serially, that is why we need **Position Embeddings** to understand the order of tokens. These embeddings insert information regarding the place of each token in the sequence, which is important for the interpretation of texts [11].

LLMs, such as *GPT*, *LLaMA* and*PaLM*, are based on architectures of Transformers with billions of parameters, thus offering advanced abilities of language comprehension and production. These models are trained on large amounts of data which renders them capable of adapting to various language tasks [10]. However, depending on their application, (that is if they focus on comprehension vs production), the models may use either part of the encoder or part of the decoder. For instance, BERT which is designed for comprehension issues, makes use only of the encoder to understand the frame and the word relations [13]. On the contrary, GPT use only the decoder because it is designed to produce text and make continuous predictions of the following words.

Pre-training

Pre-training of LLMs is implemented on huge databases of texts, without self-supervised learning, with the goal to learn the general rules of language and the relations between words and senses. During this process, the models are trying to predict which will be the following words in sentences or phrases, reinforcing in this way the comprehension of the general frame and the relations between words and sentences [13].

Fine-tuning

The process of **fine-tuning** concerns the model's adaptation to specific tasks or data. In this stage, the pre-trained model is improving in order to achieve better performance on specific applications, such as answering questions, indexing of texts, or translation. The fine-tuning can be implemented by providing examples to the model or guiding it for the specific tasks we want to execute [9].

Scaling

Scaling is one of the most important characteristics of LLMs. Given that the size of models increases, with more parameters and bigger amounts of data training, their performance

improves dramatically. The scale increase leads not only to better performance in existing tasks but also allows for new capacities to be developed, such as zero-shot learning and in general helps their improvement in more complex tasks.

Therefore, the scale of LLMs is in fact related to their performance. For instance, GPT-3 with 175 billions of parameters, surpasses other smaller models in many NLP tasks [9]. However, as models continue to develop, we notice that the more the scale increases, the more their performance continues to improve, although the profit decreases after some point [15].

The **scaling law** that arises from research work on the scale of models show that the increase of parameters and the amount of training data lead to considerable increase of performance, even when the initial improvements are small [16].

Transfer Learning

The term **Transfer Learning** refers to a model's ability to use the knowledge acquired by some tasks and to apply it to new, unknown tasks, without requiring full training from the beginning. In LLMs, transfer learning is realized with pre-training on large databases of texts and fine-tuning on smaller, more specific tasks. In this way, the model will be able to learn general rules and structures of language during the pre-training and apply them in various tasks, such as sentiment analysis, translation and question answering[13].

2.2.10 Existing LLMs and Their Characteristics

The development of LLMs has led to a wide range of approaches and architectures. In today's NLP landscape, LLMs are distinguished both in terms of purpose and characteristics.

Some models, such as **BERT** and **T5**, are primarily designed for contextual understanding and processing through pre-defined tasks (pre-trained language models). In contrast, models like **GPT-3**, **ChatGPT** and **LLaMA** place greater emphasis on generating coherent text, achieving high performance in general language generation tasks.

At the same time, more specialized models such as **PaLM**, **Minerva** and **Galactica** focus on multimodality, scientific knowledge and solving domain-specific problems. Each model, therefore, carries specific features depending on the needs of the application it targets [17].

This diversity is clearly illustrated in Figure 2.3, which presents the main categories of language modeling—from traditional statistical models to modern large language models. As

shown, LLMs now occupy a central role in both research and applications, supporting a wide range of tasks such as text generation, sentiment analysis and automatic summarization.

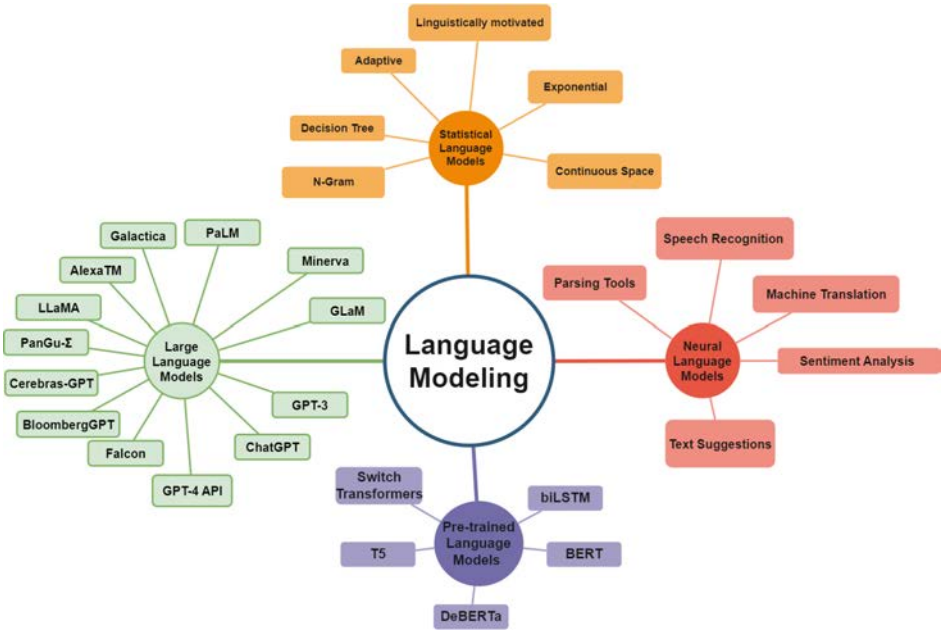

Figure 2.3: Categories and characteristics of language modeling approaches, ranging from statistical models to large language models (LLMs), along with their main applications. Image courtesy of *A Survey on Large Language Models: Applications, Challenges, Limitations and Practical Usage*.

The ongoing evolution of LLMs is expected to further enhance their capabilities in more complex tasks, such as marginalia generation and content indexing—areas of particular relevance within the context of this thesis.

2.3 Indexing

According to the author Nancy C. Mulvany and her work [18], indexing is the process of creating an index of printed works, such as books or other publications, so that readers can easily find and access information. Essentially, the index consists of an organized list of terms, ideas, or topics that appear in the text and is connected to the corresponding pages or sections of the work. This process allows users to find easily and accurately information that interests them, without having to read the whole work. The right implementation of this process requires indexing to include the content analysis of the text, highlight of the basic ideas and the creation of an analytic and full index, which is usually organized in alphabetical

order. Index quality is crucial for an effective search and has an important effect on user satisfaction. The process of index creation requires careful analysis and selection of appropriate terms, as well as resolution of issues such as synonym recognition or correct categorization of ideas. These practices ensure that the index highlights accurately the basic concepts of the work, thus offering users a powerful tool for information retrieval. In addition, the concept of indexing closely relates to the major information of a work, such as the themes and the ideas that characterize it, which are important to identify and organize in a way that facilitates a fast and accurate search. In the current era of digital information, indexing is even more important given that the modern search processes and the electronic databases are based on the correct organization and categorization of information in order to be retrieved by users. The combination of new technologies, such as LLMs, opened new horizons to improve indexing, as it allows the analysis and recognition of concepts with better accuracy and effectiveness. This technological advancement has immediate effects on the way we organize and search for information, opening new paths for the improvement of traditional methods of searching and cataloging information.

2.4 Summarization

Summarization is the process of condensing a text in order to convey its main ideas in a concise and comprehensible manner. Within the scope of traditional approaches, summarization was either performed manually by the author of the text (e.g., in the form of abstracts), or through the use of simple statistical methods such as TF-IDF (Term Frequency–Inverse Document Frequency), lexical chains and n-gram frequency, aiming to identify and extract the most representative sentences [19, 20].

The emergence of LLMs has enabled the application of advanced techniques such as abstractive summarization, where the summary is not based merely on excerpts from the original text, but is generated anew using natural and concise language [21]. These capabilities are applied in content indexing, learning platforms and the creation of customized chapter summaries.

2.5 Marginalia

Marginalia are short phrases or keywords placed in the margin of a page or screen, intended to indicate the topic or main idea of each paragraph or section. This practice has been used for centuries in manuscripts, theological and philosophical texts and in more recent publications it is incorporated to facilitate reader navigation [22].

Traditionally, marginal headings or titles are added manually by authors or editors. However, with the use of natural language models, it is now possible to automatically extract thematic terms from the text, which serve as dynamic marginal headings [23]. Such a system can support book indexing, making the structure more transparent and reader-friendly.

2.6 Questions & Answers

The generation of *questions and answers* (Q&A) is an important tool for text comprehension. Traditionally, questions are designed by the author and included at the end of each chapter or section, aiming either at self-assessment or knowledge revision.

With the advancement of large language models, it is now possible to generate questions automatically, adapted to the content and context of the paragraph or document [24]. These mechanisms, based on techniques such as retrieval-augmented generation, allow for the creation of context-aware Q&A pairs [25], which can be integrated into digital books and educational environments as tools for deeper engagement [26].

Chapter 3

State of the Art

3.1 Introduction

In the present chapter, we will explore the current development in the fields of book indexing, marginalia creation and systems of question answering (QA). The fields above constantly evolve at a rapid pace. Nowadays, new technologies offer automated and effective solutions for tasks that were up to now manual and time consuming. However, there still exist important challenges, especially, with regards to facing bias that may be included in the data or the algorithms, the representation of all opinions and the integration of various types of bibliography/references.

The chapter starts with a review of the advancements in the field of book indexing and focuses on the automation process. Next, we analyze how we can now create marginalia and we examine the application of LLMs to QA systems, highlighting their strengths and weaknesses. What is more, we will discuss various issues of software that influence the current solutions. We will thus present a complete description of the current technologies and will lay out the foundation for the resolution of further issues in the field.

3.2 Book Indexing Technology

The process of book indexing has developed considerably throughout the years. We have transitioned from traditional, manual indexing to the use of NLP and ML techniques which allowed the automation of basic processes. Nowadays, with the emergence of LLMs, indexing, marginalia creation and question answering systems are the new leaders in deeper

comprehension and more effective text processing.

In this framework, the automated book indexing combines NLP and ML for the effective creation of thematic headings. The use of LLMs in this field is a recent development, offering opportunities for deeper text comprehension and creation of rich semantic content.

3.2.1 Automated Indexing

The development of NLP and ML has revolutionized the field of book indexing, which used to be a considerably time-consuming process. The automated systems are now utilizing algorithms in order to create effective indexes, minimizing the time and effort that was required in the past. More specifically, they utilize techniques, such as keyword extraction, text summary and theme modeling. Methods, such as supervised machine learning with labeled training data, turned out to be successful in improving indexing accuracy [27]. These methods allow systems to find and classify keywords with minimal human intervention.

A notable development relative to the use of algorithms of theme modeling is Latent Dirichlet Allocation (LDA) which finds basic themes inside texts[28]. Nevertheless, there are still challenges for the creation of automated indexes of high quality, such as how to treat polysemy of words and the depth of terms, which are important obstacles in obtaining quality that attains human accuracy [27].

3.2.2 Automated Indexing - Current Technological Approaches

The current indexing techniques are based on two basic approaches:

1. **Lexical Approaches**: they include algorithms that are based on correlation of keywords in texts with terms from a controlled lexicon. Examples include:
    - *KEA και Maui*: They are used for subject indexing via matching of words in lexicons, such as the Medical Subject Headings [29].
    - *TF-IDF*: It calculates the meaning of terms in a body of texts via frequencies [30].
2. **Machine Learning Approaches**: They are based on the use of large bodies of data for model training that predict thematic headings. Important algorithms include:
    - *FastXML*: It specializes in tree-based classification [31].

- *fastText*: It offers fast training and high accuracy in classifications [32].
- *Vowpal Wabbit (VW)*: It uses methods of online learning for continuous improvement of sentences [33].

The above theoretical classification in lexical and machine approaches is applied in various ways on modern tools and systems. Many of these have been developed for subject indexing of documents, nevertheless, some of them focus or can be expanded in analytic book indexing. In what follows, we present the representative applications that put the above technologies into action.

3.2.3 Tools for Book Indexing

The technological progress in the field of automated book indexing has led to the development of a plethora of tools, commercial and open access. The following tools are utilized largely for the creation of "back-of-the-book" indexes and they present different levels of automation and adaptation.

- **TExtract**: Commercial tool that supports the automated extraction and processing of key words in documents in PDF and Word format. It provides an index that is capable of extraction in multiple formsς [34].
- **PDF Index Generator**: It specializes in indexing of PDF documents, using keywords and phrases that are located automatically. It supports interactive processing of the index by the user. [35].
- **Picardy**: This is a free software for the creation, processing and formatting of indexes in Windows, Linux and macOS. It supports extraction in LaTex [36].
- **Cindex**: Professional tool that supports multiple forms of extraction of indexes (Word, RTF, HTML, XML) and provides advanced processing functions. It is available now as an open access software [37].
- **Xindy**: It is used mostly for documents in LaTex, with Unicode support and multilingual indexing [38].

The tools above are included in the category of traditional indexing systems which are based on localization techniques of keywords and phrases via rules, language processing

(e.g. regular expressions, syntactic matching, stemming) and frequency of use. They do not use artificial intelligence techniques or semantic analysis and therefore, they do not result in content comprehension based on context.

Integration of Artificial Intelligence in Indexing

The use of artificial intelligence speeds up the process of indexing via conceptual comprehension and classification. Some important research attempts include:

- **Kratt**: A platform that was developed by the National Library of Estonia for thematic book indexing. It applies NLP and supervised learning for automated assignment of thematic terms [39].
- **Generative Book Search (GBS)**: It proposes a framework of indexing via genetic models and use of outline+main text combined. It improves 9.8% the performance in retrieval tasks [40].

Among the indexing tools that have been largely evaluated in national libraries and digital infrastructure, Annif is distinguished due to its multialgorithm architecture and its capacity to adapt in various thematic environments. For this reason, it is presented separately below.

Annif: A Multialgorithm Tool of Subject Indexing

Annif is an open access tool for automated subject indexing of documents, developed by the National Library of Finland [41]. Its architecture is based on a multialgorithm approach (ensemble learning), combining machine learning techniques and NLP for the evaluation and assignment of subject terms to documents.

Annif supports parallel or consecutive use of different algorithms such as:

- **TF-IDF**, for the valuation of the relevance of terms in a body of texts,
- **fastText**, for rapid and efficient classification of subjects based on educational data,
- **Maui**, which is based on lexical matching and supervised learning [42],
- **Omikuji**, a decision tree learner appropriate for multicategory labels.

The tool enables training on specific lexicons and classification systems, such as LCSH, YSO or Medical Subject Headings (MeSH), making it particularly flexible in adapting to different thematic environments.

Although it was not designed initially for analytic book indexing, Annif can be adapted to this type of applications. The input of custom vocabularies and the ability for local training make it an appropriate candidate for use in environments, such as libraries, digital archives and publishers' platforms, where there is a high need for thematic coverage at the level of sections/units.

Nevertheless, despite its flexible architecture, Annif is based mostly on traditional machine learning techniques. This means that its ability to understand more abstract or deep semantic structures is restricted, something that is crucial for the indexing of large and multidisciplinary books. In order to strengthen its semantic abilities, research attempts have already started to incorporate LLMs into the architecture of Annif [43].

Overall, Annif constitutes the most complete, actively supported and documented open access tool for the automation of subject indexing. Its architecture, in combination with its support of multiple methods and vocabularies, makes it extremely valuable for applications that require harmonized indexing in large scale collections.

3.2.4 Recent Research Approaches of Automated Book Indexing

In addition to the tools of subject indexing developed for digital archives or bibliography entries, recent research bibliography turns more to book indexing aiming for an automated index creation of the type "back-of-the-book". The approaches below combine traditional NLP techniques, semantic analysis and specialized methods of extraction of terms.

TMG-BoBI. *TMG-BoBI* (Text-to-Matrix Generator for Back-of-the-Book Indexing) is proposed by Koutropoulou and Gallopoulos (2023) and is based on the known tool TMG. The system utilizes a series of techniques for index creation:

- Automatic extraction of keywords (unigrams, bigrams, trigrams)[1],
- POS tagging for finding substantive and characteristic phrasal patterns[2],

[1]Unigram: a word (e.g. *indexing*), Bigram: two consecutive words (e.g. *subject indexing*), Trigram: three consecutive words (e.g. *automated subject indexing*).

[2]Part-of-Speech tagging: identification of the grammatical category of each word in the text (e.g. *noun*, *verb*, *adjective*).

- Analysis of keywords position[3],
- Index creation in compatible form of LaTeX via MakeIndex[4].

TMG-BoBI allows regulation of the index's exhaustion level via the parameter `IndexProb` and shows improved performance compared to other tools, such as TExtract. Although it requires use of MATLAB and LaTeX, it has an important contribution to the complete automation of analytic indexing [44].

Terminology/Keyphrase Extraction for Creation of Book Indexes in Polish. Marciniak, Mykowiecka and Rychlik's approach focuses on the automated index creation in books written in Polish. The system is based on the extraction of conceptual terms and keyphrases, via:

- Morphosyntactic analysis[5],
- Lemmatization and localization of verbose expressions[6],
- Filtering based on frequency and semantic relevance[7].

The study highlights the significance of language adaptation in indexing, avoiding models of "one-size-fits-all" and putting emphasis on the grammar and syntactic structure of every language [45].

Back-of-the-Book Index Automation for Arabic Documents. Haidar and Zaraket's (2004) work presents a completely automated indexing system of documents in Arabic. Their approach is based on terms extraction and identification of their place in the text, via:

- Extraction of keyphrases with semantic analysis[8],

[3] Evaluation of the frequency and the position in which a word surfaces in the text, such as at the *beginning*, in *titles*, or in *repetitive spots*, in order to assess its significance.

[4] *MakeIndex* is a LaTeX tool that creates automatically index tables, connecting keywords with the pages in which they appear.

[5] Analysis of sentences' *grammatical structure*, with the goal to identify the function of each word (e.g. *subject*, *verb*, *object*) in order to find useful phrases for indexing.

[6] Lemmatization: conversion of words into their basic form (e.g. *trehondas* → *treho*). *Verbose expressions*: phrases of two or more words that have a single meaning (e.g. *artificial intelligence*)..

[7] *Frequency* checks how often a term appears in a text, whereas *semantic relevance* evaluates if it is related essentially with the text meaning.

[8] *Semantic analysis* allows the identification of concepts based on their meaning and not just their word form.

- Computation of cosine similarity [9] using vector embeddings[10],
- Index creation in JSON form with metadata about place and relevance[11].

The system has achieved a very high F1-score (0.966) and has proved that the semantic approach can function effectively in languages with high morphological complexity, such as Arabic. Although it does not incorporate LLMs, its architecture makes it appropriate for future incorporation [46].

Summary. The analysis of the approaches presented above shows the transition from traditional subject indexing tools to specialized methods of analytic book index creation. Despite the different target language and technical implementation, the common goal is the utilization of natural language techniques and the need for adaptation, as far as language and type of content are concerned.

3.2.5 Creation of Marginalia Using LLMs

Marginal titles, as succinct phrases placed in the margin or body of a text, constitute a form of selective summary. The process of their creation relates closely to techniques of abstractive summarization, as it requires the condensation of content in few indicative words.

In addition, marginal titles can be considered a type of "marginalia" – marginal comments or indications that summarize locally the content of a text and function as navigation and comprehension guides for the user. According to recent studies, the use of LLMs for the creation of succinct annotations, either as marginalia or as digital marginalia, is based on common mechanisms of semantic extraction and simulates the process of summative representation of meaning [47].

The use of LLMs in the process of indexing has brought significant changes, contributing to the automated creation of marginalia. Marginalia constitute a crucial element of indexing as they assist readers to navigate easier in any given book and to find relevant information in large texts.

[9] *Metric* that compares how similar two phrases or concepts are, depending on the angle of their vector in multidimensional space.

[10] *Vector embeddings* are numerical representations of words or phrases that hold semantic information.

[11] *JSON* is a form of structured data that allows storage of place (e.g. *page*, *paragraph*) and relevance (e.g. *semantic closeness*) of each term.

The Process of Marginalia Creation by LLMs

The automated creation of marginalia via LLMs has proved to be a promising technology for text processing and marginalia creation. Compared to human work, it seems that LLMs can create titles of high quality, but they often exhibit limitations relative to the accuracy and completeness of content [48].

More specifically, LLMs, such as ChatGPT and BERT, can analyze big texts, as they can find subject units and thus, they propose marginalia based on concepts that they extract from the data. The process includes the following steps:

- **Text Analysis:** The models examine the text and find subject units using semantic comprehension techniques.
- **Titles Creation:** Based on the analysis, LLMs create marginalia that summarize the content of each unit.
- **Improvement:** The marginalia are being adapted so that they are brief, concise and representative of the content.

Although their function is obvious, LLMs face challenges, such difficulty in comprehending complex concepts or creating content that can bring about cultural and social biases. All of the above may affect the accuracy of marginalia.

Challenges and Limitations

In general, the use of LLMs and their implementation in marginalia creation raises some issues:

- **Superficial Comprehension:** LLMs can create marginalia that do not reflect the actual complexity of the text [49].
- **Cultural Biases:** Marginalia created by LLMs may not be neutral, incorporating cultural or social biases that exist in the training data[50].
- **Treatment of Special Contents:** Marginalia creation for specialized or technical texts may be less effective due to the lack of the model's special training [6].

Improvements and Prospects

The following approaches have been proposed to face challenges discussed above:

- **Adaptation to Local Data:** LLMs training with local data can reduce biases.
- **Combination of Human Intervention:** The collaboration of LLMs and human curators can ensure accuracy and quality in marginalia [51].

In conclusion, marginalia creation from large language models is a very promising tools for books, but more studies are required for the resolution of their limitations and improvement of their reliability.

3.3 LLMs in Question Answering Systems

3.3.1 LLMs Capacities in QA

Recently, the field of Question Answering Systems has been transformed by large language models, such as GPT-4 and BERT, utilizing extended pre-training in various data sets. These models prevail in comprehension of frames, creation of coherent answers and handling of open-ended questions [13]. Their answers are conscious of the frame and this makes them valuable for application in education, customer support and research. LLMs are also notably capable of managing multilingual and multimodal questions. For instance, GPT-4 has been proven sufficient to process questions that combine text and images. Additionally, there is evidence that LLMs' refinement in datasets of specific fields can improve the accuracy and relevance in these specialized fields [52].

3.3.2 Limits and Challenges

As we have seen, LLMs exhibit many advantages. This does not mean that they do not face important challenges. More specifically, one of the most crucial issues is their sensitivity in creating "illusions" or inaccurate answers that seem plausible [53]. This outcome undermines their reliability.

Another challenge concerns the models' lack of 'justice' given that data may include biases. These biases may reinforce stereotypes or exclude marginal perspectives [49].Attempts

to mitigate these biases include the development of training methods with 'justice' awareness and the incorporation of different data sets [54].

Finally, LLMs seem to have difficulties with reasoning on long contexts or very complex questions. The progress seen in models' architectures, such as neural networks with reinforced memory, aim to resolve these limitations [55].

3.3.3 Typical Issues

Gradation and Accuracy The volume of data that we have to manage is often very large and results in performance and accuracy issues. Most tools face difficulties in simultaneous management of multi-source data, as well as production of reliable results [51].

Incorporation with Existing Systems Legacy systems have often difficulty in collaborating with modern indexing and question answering tools, causing problems in the transition.

Management of Various Forms of Input Input forms vary. There are, for instance, PDF documents, websites and unstructured text. As a result, the processing of such input from current software becomes difficult and we may not receive the optimal results[6].

3.3.4 Improvement Prospects

These issues can be addressed via:

- **Development of tools for more efficient gradation**
- **Creation of common standards for the interconnection of systems**
- **Development of algorithms capable of processing diverse data**

3.4 Conclusion

In this chapter, we examined the State of the Art in the domain of indexing, creation of marginalia and question answering systems. We pointed out that there has been considerable progress in the automation via LLMs, as well as problems with regards to marginalia creation.

The attested issues that are still pervasive, such as biases, gradation and difficulty of integration with existing systems, require immediate attention and response. Failing to address these issues may limit the effectiveness and justice of the systems.

Chapter 4

Specifications and User Needs

4.1 Introduction

Understanding user needs and clearly defining the specifications are critical steps in any system development project. The proposed system for automatic indexing and the generation of marginal headings and question–answer pairs aims to address essential requirements arising both from the perspective of book production and editing, as well as from their educational use.

4.2 User Groups and Needs

The primary users of the system can be divided into two main categories.

4.2.1 Authors and Publishers

Primarily, the system is intended for authors and publishers. Although most books include an index, its creation—along with the development of accompanying material such as marginal headings and question–answer pairs—is often a particularly time-consuming and demanding task. The proposed system offers an automated and faster method for generating such material, enhancing the workflow and reducing the need for manual effort. At the same time, it enables the enrichment of the book's content by providing structured summaries and comprehension questions, which enhance its educational value and facilitate its use in learning environments.

4.2.2 Students and Educators

Secondarily, the system addresses the needs of students, academics and educators. These users seek efficient ways to comprehend and review extensive scientific or theoretical texts. Through the use of marginal headings, summaries and question–answer pairs, the system supports quicker navigation through the book and reinforces the process of active learning.

4.3 System Specifications

The developed system meets the following core specifications:

File Input: Ability to read and process Greek-language PDF books. In its current version, the system exclusively supports PDF files. However, due to the nature of the processing—which is based on plain text—extending support to additional file formats such as `.txt` and `.docx` is considered feasible and is suggested as a future improvement.

The remaining core functionalities are summarized as follows:

- **Content analysis:** Division of the text into smaller parts (paragraphs and sections).
- **Extraction of key concepts:** Identification and extraction of the main concepts from each paragraph, for the purpose of index creation.
- **Summarization and titles:** Generation of a short summary and a representative marginal heading for each paragraph.
- **Section summaries:** Creation of a concise summary for each section of the document.
- **Questions and answers:** Automatic generation of two question–answer pairs per section.
- **Output storage:** Export of the generated data in CSV and TXT format.

The implementation of the above functionalities relies on the use of `meltemi_q8`, a quantized (8-bit) version of the *Meltemi-7B-Instruct v1.5* LLM, specifically fine-tuned for the Greek language. This model is accessed locally via the Ollama API, offering full independence from cloud-based services and ensuring enhanced privacy and autonomy during execution.

4.4 Limitations and Assumptions

The system assumes that the input files contain clean and readable text, free from optical character recognition (OCR) errors and that they are written in the Greek language. Additionally, it focuses solely on text analysis and does not provide a graphical user interface (GUI), operating exclusively through the command line interface (CLI). The quality of the generated information directly depends on the linguistic and conceptual capabilities of the model used, *Meltemi-7B-Instruct v1.5*.

4.5 Use Cases

Indicatively, the system may be used in the following ways:

- A publisher imports the PDF of a book and automatically receives an index, marginal headings and question–answer pairs that can be integrated into the final edition.
- A student uses the summaries and Q&A to support faster comprehension and review of the material.

4.6 Conclusions

In summary, the system developed aims to bridge the gap between book production and their educational use. On one hand, it facilitates the work of authors and publishers by providing a faster and more efficient way to generate supporting material. On the other hand, it enhances the experience of the end user by offering tools for better understanding, navigation and knowledge revision.

Chapter 5

Implementations

5.1 System Implementation

This chapter outlines in detail the individual subsystems developed for the automated processing of the book's text. The aim was to create an index, generate marginal headings for each paragraph and produce question–answer sets for each thematic section. The system's architecture follows a modular structure, consisting of four distinct phases:

- Extraction of key concepts from paragraphs for indexing
- Generation of headings (marginalia) based on each paragraph's summary
- Creation of concise summaries for each thematic section
- Generation of comprehension question–answer pairs per section

Each subsystem was implemented in `Python` and uses the local language model `meltemi_q8` via HTTP API (`Ollama`). Several libraries were used, including `requests`, `re`, `csv`, `psutil`, `platform`, `time`, `json` and `PyPDF2` for PDF document handling.

The following sections provide a detailed overview of each subsystem's structure and functionality, including the prompts used and the performance metrics.

5.2 Data Source and Text Processing

The source material is the Greek-language legal textbook titled *«Προσωπικές Εταιρείες»*, available in PDF format. The analysis focused on the book's thematic structure (paragraphs and sections), regardless of pagination.

Raw text extraction was performed using a custom Python parser that reads the PDF content page by page, removes footnotes and artificial separators and segments the content into coherent paragraphs or page bundles, depending on each subsystem's requirements. At the same time, each excerpt remains linked to its corresponding original page numbers.

The resulting text format is used across all subsystems, such as indexing, summarization and Q&A generation. To ensure smooth communication with the language models, additional stages of tokenization, encoding and decoding are applied, as described in the sections that follow.

Tokenization, Encoding and Decoding via Ollama

The processes of tokenization, encoding and decoding were carried out via the Ollama API, which serves as an intermediary communication layer with the local LLMs. In more detail:

- Each text chunk is sent to the Ollama API, where automatic tokenization is applied based on the selected model.
- Before the prompt is sent, a size check ensures that the prompt does not exceed the token limit supported by the model (e.g., 4000 tokens).
- During the encoding phase, the model transforms the tokens into an intermediate semantic representation in order to generate a response.
- The output is returned as decoded text, which is then used for further processing (e.g., title generation, concept extraction, summarization).

This process is handled transparently for the user, requiring no manual management of tokens or encoding parameters and offers flexibility and compatibility across different models.

5.3 Key Concept Indexing Subsystem

Subsystem Objective

The purpose of this subsystem is the automatic extraction and indexing of the key concepts identified in the text body. These concepts are recorded along with the pages on which

they appear, forming a hierarchical index of keywords covering the entire content.

Pipeline Description

This process is implemented via the `GreekSemanticChunker` class and includes the following stages:

1. **Paragraph Extraction:** The PDF file is broken into paragraphs (chunks) using the `extract_text_from_pdf()` function, which detects actual paragraph structure, removes footnotes and maintains a mapping to the original page numbers.

2. **Content Summarization:** Although the key concepts are extracted from the original text, each paragraph is also summarized briefly. The summary serves as a cleaner version of the content and may be used in the future to extract more thematically focused concepts.

3. **Key Concept Extraction:** For each chunk, a separate prompt is sent to the LLM in order to retrieve 3–5 core concepts.

   Prompt (translated in English):

   > **Text:** [chunk]
   >
   > Extract the 3–5 most important concepts from the above text.
   > Write only the concepts, without numbering, prefixes or explanations.
   > Avoid full sentences or concepts consisting of a single character.
   > Write one concept per line.

4. **Cleaning and Filtering:** Extracted concepts are cleaned of numbers, bullet points, or unnecessary words. Lines containing redundant meta-language (e.g., “concepts:”, “please”, “answer:”) are ignored.

5. **Index Construction:** Concepts are linked to the pages on which they appear:

   - Single-word concepts are recorded directly as index entries.
   - Multi-word concepts (e.g., “commercial company”) are stored hierarchically, e.g., `”commercial”: ”company”: [12,13]`.

6. **Result Storage:** Concepts and summaries are stored per chunk, while the final index is constructed as a dictionary during execution and exported as formatted text in the files `output_<model>.txt` and `index.txt`. For use in other systems, it can easily be converted to `JSON` format.

Execution Statistics

The processing was applied to the entire book (1308 chunks). The system on which it was executed had the following specifications:

- **Platform:** Linux 5.15.0-131-generic (x86_64)
- **Processor:** 56 physical cores, 112 logical
- **RAM:** 188.08 GB (98.19 GB available)
- **Model:** `meltemi_q8`
- **Total processing time:** 536 minutes and 54.46 seconds
- **Average duration per chunk:** 24.63 seconds
- **Memory usage:** 37.46 MB

5.4 Marginalia Generation Subsystem

Subsystem Objective

The goal of this subsystem is to generate short, semantically accurate titles—marginalia—for each thematic unit (chunk) of the text. These subheadings enhance navigation, readability and the overall structure of the content, acting as thematic markers.

Pipeline Description

The marginal heading generation process is integrated into the `GreekSemanticChunker` and is based on the summaries produced in the previous stage. It consists of the following steps:

1. **Summary Generation:** A new concise summary is generated for each chunk, serving as input for the title generation.

2. **Prompt Creation:** Each summary is transformed into a prompt, instructing the model to produce a title of fewer than 10 words, without explanations, punctuation, or introductory phrases.

3. **API Call:** The prompt is sent to the local `meltemi_q8` model via the Ollama API and the response is cleaned of unnecessary characters.

4. **Result Storage:** For each chunk, the generated title is stored along with the corresponding summary and key text statistics in the file `output_headings.txt`.

Prompt (translated in English):

> **Summary:** [summarized text]
>
> Write a short and concise title for the above text.
> The title must be under 10 words, with no introductory phrases or explanations.

Execution Statistics

Marginalia generation was applied to the entire book (1308 chunks), using the same local model as in the indexing subsystem.

- **Platform:** Linux 5.15.0-131-generic (x86_64)
- **Processor:** 56 physical cores, 112 logical
- **RAM:** 188.08 GB (97.79 GB available)
- **Model:** `meltemi_q8:latest`
- **Total processing time:** 497 minutes and 37.08 seconds
- **Average duration per chunk:** 22.83 seconds
- **Memory usage:** 38.11 MB
- **Total titles generated:** 1308

5.5 Subsystem for Generating Section Summaries

Subsystem Objective

This subsystem aims to automatically generate concise summaries for each thematic section of the book. These summaries serve both as a helpful tool for quickly understanding the book's content and as an intermediate input for the next stage of the pipeline—question–answer generation.

Pipeline Description

The implementation was done in Python and consists of the following stages:

1. **Section Definition:** The book was divided into 20 logical sections using a dictionary (`SECTIONS`) that maps each section to a specific range of PDF pages.

2. **Text Extraction:** For each section, the relevant text is extracted using the `extract_chapter_chunks()` function and stored as a list of paragraphs, which are then joined into a single text body (`full_text`).

3. **Prompt Creation and LLM Call:** Each `full_text` is paired with a prompt and sent to the local language model (`meltemi_q8`) via HTTP POST request to the local API.

4. **Handling Long Texts:** If the full text exceeds 10,000 characters, it is split into segments of 4,000 characters. A separate summary is generated for each segment and the final output is the concatenation of these partial summaries.

5. **Error Detection:** The model's output is checked for words such as `error`, in order to flag potentially problematic sections.

6. **Result Storage:** The results are saved in the following files:

   - `section_summaries.csv`
   - `section_summaries.txt`
   - `processing_report.txt`

Prompt (translated in English):

> Provide a short and concise summary for the following text:
>
> [section text]
>
> **Summary:**

Execution Statistics

During the execution of the subsystem, summaries were generated for 20 sections. Below are the system specifications and summary statistics:

- **Platform:** Linux 5.15.0-136-generic (x86_64)
- **Processor:** 56 physical cores, 112 logical
- **RAM:** 188.08 GB (165.1 GB available)
- **Model:** `meltemi_q8`
- **Total processing time:** 279 minutes and 10.37 seconds
- **Average duration per section:** 835.52 seconds
- **Memory usage:** 8425.57 MB

5.6 Subsystem for Generating Section-Based Question–Answer Pairs

Subsystem Objective

The purpose of this subsystem is to automatically generate two question–answer pairs for each thematic section of the book, based on the corresponding summary produced in the previous stage. These Q&A pairs aim to reinforce content understanding and serve as a useful tool for revision and self-assessment.

Pipeline Description

The process consists of the following steps:

1. **Summary Cleaning:** Unrelated or generic phrases often added by language models (e.g., “I am a model…”, “my role is…”) are removed from the input.

2. **Prompt Creation:** Based on the summary, a prompt is created to request the generation of two question–answer pairs in a strict format. If the prompt exceeds 4000 characters, it is truncated to fit the model’s input limit, prioritizing the preservation of the most important and introductory parts of the summary.

3. **API Call:** The prompt is sent via HTTP POST to the local model API (Ollama). If the request fails, a retry mechanism is triggered, allowing up to two attempts.

4. **Output Parsing:** The model’s response is processed using regular expressions (regex) to identify question and answer pairs. The subsystem retains up to two valid pairs.

5. **Result Storage:** The results are saved in the following files:

   - `qna_from_summaries.csv`
   - `qna_from_summaries.txt`
   - `qna_report.txt`

Prompt (translated in English):

Summary: [cleaned summary]

Based on the above summary, write two questions and their answers.
Use **EXACTLY** the following format:

Question 1:
Answer 1:

Question 2:
Answer 2:

Execution Statistics

Question–answer generation was carried out for all sections of the book, based on their corresponding summaries. The process was completed successfully with minimal resource usage. Below are the main system specifications and processing statistics:

- **Platform:** Linux 5.15.0-136-generic (x86_64)
- **Processor:** 56 physical cores, 112 logical
- **RAM:** 188.08 GB (165.21 GB available)
- **Execution time:** 13 minutes and 58.9 seconds
- **Memory usage during execution:** 1.09 MB
- **Number of sections:** 20

5.7 Implementation Conclusions

The implementation of the four main subsystems highlighted the potential of using large language models for processing Greek-language texts with legal terminology—aiming at conceptual restructuring, thematic summarization and supporting comprehension. Each subsystem serves a distinct role but integrates seamlessly with the others, allowing the text to be automatically processed from initial extraction to deeper understanding.

Through indexing and marginalia generation, the text was organized not merely by its formal structure (e.g., pages or chapters), but according to its meaning and the key concepts it contains. At the same time, the creation of summaries and question–answer pairs added educational functionality to the system, making the content more accessible and reusable.

Although the results are encouraging, output quality varies and depends on factors such as the style of the original text, the density of information, the clarity of the prompts and the inherent difficulty of processing the Greek language with LLMs. Therefore, systematic evaluation is necessary to determine the functional adequacy of the subsystems.

The next chapter is dedicated to the evaluation of the generated metadata and outputs, through quantitative metrics such as ROUGE and BERTScore, as well as qualitative criteria, preparing the ground for future improvement and expansion of the system.

Chapter 6

Evaluation

6.1 Introduction

The evaluation of the results obtained during the execution of the experiments constitutes an important part of this work, as it allows us to assess their quality and reliability. The purpose of this process is to examine the accuracy, conciseness and semantic consistency of the generated index, the marginal headings in relation to the summaries produced and the question–answer pairs based on the text. Through this analysis, we will identify areas that need improvement, while also documenting the effectiveness of the methods used.

6.2 Evaluation of the Automated Index

6.2.1 Evaluation Methodology

To assess the quality of the automatically generated index, we used the book's existing expert-created index as ground truth. The evaluation focused exclusively on comparing the terms (concepts), without considering page numbers. The process followed these steps:

- **Term Normalization**: All terms were converted to lowercase, accents were removed and common Greek stopwords were filtered out, in order to reduce the impact of superficial linguistic differences on matching.
- **Fuzzy Matching**: Partial term matching was applied using the `difflib.get_close_matches()` function with a similarity threshold of **0.7**, to identify terms with minor variations (e.g., inflectional differences or changes in word

order). This threshold was chosen as a balanced value—lenient enough to allow slight variations but strict enough to avoid overly loose matches.

- **Metric Calculation**: The metrics **Precision**, **Recall** and **F1 Score** were used to evaluate the performance of the automated index in comparison with the manually curated one.

Precision reflects the proportion of correctly identified terms out of all terms proposed by the system. Recall measures how many of the terms from the manual index were successfully retrieved. Finally, the F1 Score provides an overall performance indicator by combining both precision and recall into a single value.

6.2.2 Evaluation Results

The evaluation produced the following results:

Metric	Value
Precision	25.99%
Recall	36.98%
F1 Score	30.52%

Table 6.1: Evaluation results of the automated index

Interpretation: The results indicate that the system is able to identify a significant portion of the terms found in the manually created index (Recall: 36.98%), but also includes a considerable number of terms that do not correspond to actual concepts (Precision: 25.99%). The combined metric F1 Score (30.52%) reflects an overall moderate system performance, highlighting room for improvement in both precision and recall.

6.2.3 Visualization of Evaluation Results

The pie chart illustrates the evaluation outcomes:

- **Correct Matches**: 18.0%
- **Missed (manual)**: 30.7%
- **Wrong (auto)**: 51.3%

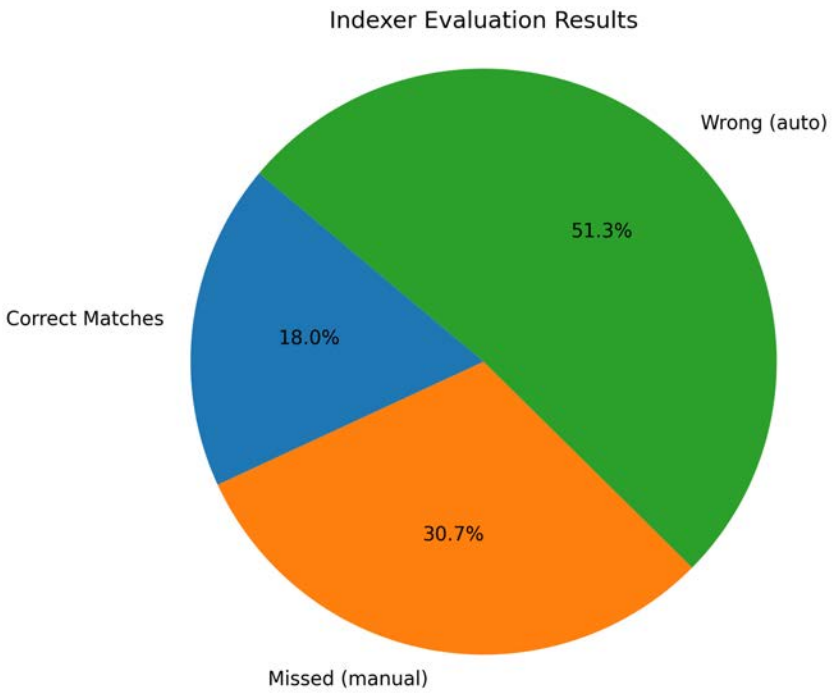

Figure 6.1: Graphical Representation of the Automated Index Evaluation

The evaluation results reveal a relatively low percentage of correct matches (18.0%) and a significantly high portion of incorrect automatic matches (51.3%). This performance can be attributed to several critical factors in the current indexing pipeline:

- **Concept Extraction Limitations:** The extraction of key concepts relies on a language model prompt that requests concise keywords. Nevertheless, the results might contain ambiguous, excessively general, or context-insensitive terms that don't exactly appear in the original paragraph, which could result in mismatches or false positives.
- **Matching Heuristics:** The system checks if the extracted concept (or its components) exist *as substrings* in the paragraph text. This method is oversimplified and ignores Greek morphological variances, synonyms, and grammatical inflections. For instance, "φορολογία" could appear as "φορολογίας" or "φορολογικό σύστημα," and these variations might not match.
- **Multi-word Concept Handling:** Concepts with multiple words are handled hierarchically, with the first word serving as the main category and the remaining words as subcategories. Matching and evaluation are made more difficult by this structure, particularly when the entire phrase is fragmented across sentences or is not consistently present in the text.

- **No Semantic Validation:** There is no semantic similarity check (e.g., using embeddings) between the extracted concept and the paragraph text. Thus, purely lexical checks are prone to both overgeneration and undergeneration errors.

In conclusion, lexical-level matching and the absence of strong concept validation mechanisms impair indexing performance, even though the current pipeline shows some ability to extract pertinent content. These findings emphasize the need for more domain-specific fine-tuning and better natural language processing methods.

6.2.4 Comment on Page Numbers

Although this evaluation focused exclusively on index terms, it is worth noting that the system also stores the page numbers where each term appears in the book.

Page association is performed in a simple yet reliable manner: during the processing of the PDF file, each extracted paragraph is linked to its exact page number. Then, when the system detects a term within a paragraph, it associates that term with the page(s) corresponding to that paragraph.

Through this process, the page numbers assigned to each term are not estimated arbitrarily; rather, they are based on the actual location of the text in the book. As a result, we can consider the page information generated by the system to be trustworthy and a precise reflection of where each term appears in the original document.

6.2.5 Evaluation of Computational Performance

The evaluation of the system's computational performance showed that, despite the large volume of data and the demanding nature of the task, the process was completed with exceptionally low memory usage and within an acceptable time frame. Full statistics were presented in the previous chapter.

These results demonstrate that the method is computationally efficient and can be applied to large-scale projects, provided that adequate infrastructure is available. It is worth noting that the entire process was applied to a 592-page legal textbook, which highlights the system's practical utility. Compared to manual indexing, the automated procedure offers significant time savings, making the tool especially valuable in real-world production settings.

6.3 Evaluation of Generated Summaries and Marginalia

To evaluate the quality of the generated summaries and marginalia, a systematic approach was applied based on two well-established metrics: BERTScore and ROUGE. These metrics were chosen to capture both the semantic similarity and the lexical overlap between the generated and the reference texts.

The evaluation consisted of two stages:

- **Summary Evaluation:** Similarity was measured between each original text (*chunk*) and its corresponding generated summary.
- **Title Evaluation:** Similarity was measured between each generated summary and the title produced from it.

BERTScore (F1) evaluates the semantic similarity between two texts by comparing the token embeddings produced by a pre-trained model. The higher the BERTScore, the greater the semantic alignment between the two texts. For this evaluation, the pre-trained model **xlm-roberta-large** was used, as it supports multiple languages, including Greek.

The *ROUGE* metric measures word overlap (unigrams) and the longest common subsequence between the candidate and reference text, providing indicators such as *ROUGE-1* and *ROUGE-L*.

6.3.1 Results

The average metric scores obtained from the evaluation are as follows:

Table 6.2: Average Evaluation Scores for Summaries and Titles

Metric	**Summary (Chunk vs Summary)**	**Title (Summary vs Title)**
BERTScore (F1)	91,46%	86,58%
ROUGE-1	10,02%	1,05%
ROUGE-L	9,95%	1,05%

Interpretation of Results

The evaluation scores for both summaries and titles offer a detailed view of the quality of the generated subheadings:

- **BERTScore (F1) – 0.9146 for summaries:** The high BERTScore indicates that the generated summaries are very close to the original text, both in terms of word embeddings and overall meaning.
- **ROUGE-1 (0.1002) and ROUGE-L (0.0995) for summaries:** Despite the high semantic similarity shown by BERTScore, the ROUGE values are low. This suggests that the summaries do not reuse many exact words from the original text, likely due to the use of synonyms or more generalized language.
- **BERTScore (F1) – 0.8658 for marginal titles:** The BERTScore for marginal titles is lower than for summaries, indicating that the titles are not as semantically close to the summaries as the summaries are to the original chunks. Nevertheless, the score remains relatively high, showing that the titles are still meaningfully related to the summaries.
- **ROUGE-1 (0.0105) and ROUGE-L (0.0105) for marginal titles:** These very low scores suggest that the titles share few exact words with the summaries and that word order differs significantly—something expected, as a title typically does not repeat the original text.

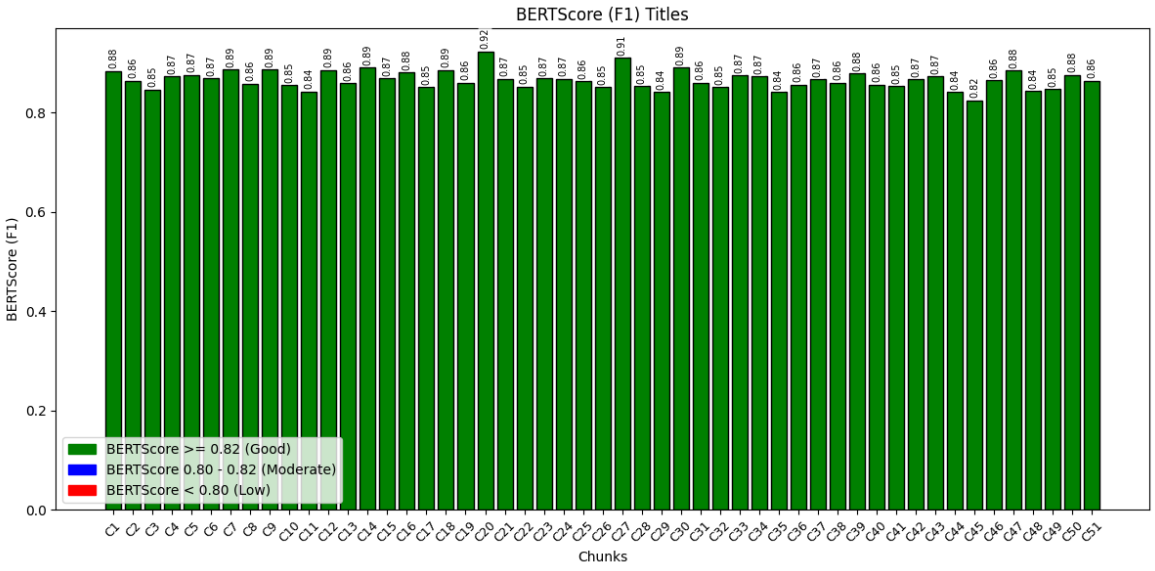

Figure 6.2: BERTScore (F1) per generated title for the first 51 text chunks

As shown in Figure 6.2, the *BERTScore* values are generally high and remain stable across all marginal titles. This consistency may indicate that the system generates titles of similar quality for all paragraphs, regardless of their content.

6.3.2 Scope of Evaluation and Computational Assessment

The evaluation of the quality of the generated summaries and subheadings was carried out on two levels:

- **Sample-based evaluation:** The evaluation was performed on 51 text chunks from the first section of the book. The *BERTScore* and *ROUGE* metrics were used to assess semantic similarity and phrasing overlap between the generated summaries and the original paragraphs.

- **Full-scale computational processing:** The summarization and marginalia generation method was applied to the entire book (1308 chunks), in order to evaluate the system's computational performance at scale. No quantitative evaluation (BERTScore, ROUGE) was performed on the full dataset, as it was limited to the sample. The processing was executed in the computational environment described in the previous chapter, with a total runtime of approximately 8.5 hours and low memory usage (38 MB).

Applying the system to the entire book demonstrates that it can operate effectively in practice and lays the groundwork for more extensive future evaluations—both with larger datasets and with more in-depth analysis of output quality.

6.4 Evaluation of Generated Section Summaries

To assess the quality of the concise summaries produced for the 20 thematic sections of the book, a quantitative evaluation was conducted using the metrics *BERTScore (F1)* and *ROUGE*.

The BERTScore values were calculated based on the semantic closeness between each generated summary and the original full text of the corresponding section. For this purpose, the pre-trained model `xlm-roberta-large` was used again via the `bert-score` library. At the same time, ROUGE-1 and ROUGE-L scores were computed to estimate surface-level lexical and phrase overlap.

Figure 6.3 presents the system's performance in terms of BERTScore (F1) across the sections. Sections with a score ≥ 0.82 were considered *"good"*, those between 0.80–0.82 were labeled *"moderate"* and no section scored below 0.80.

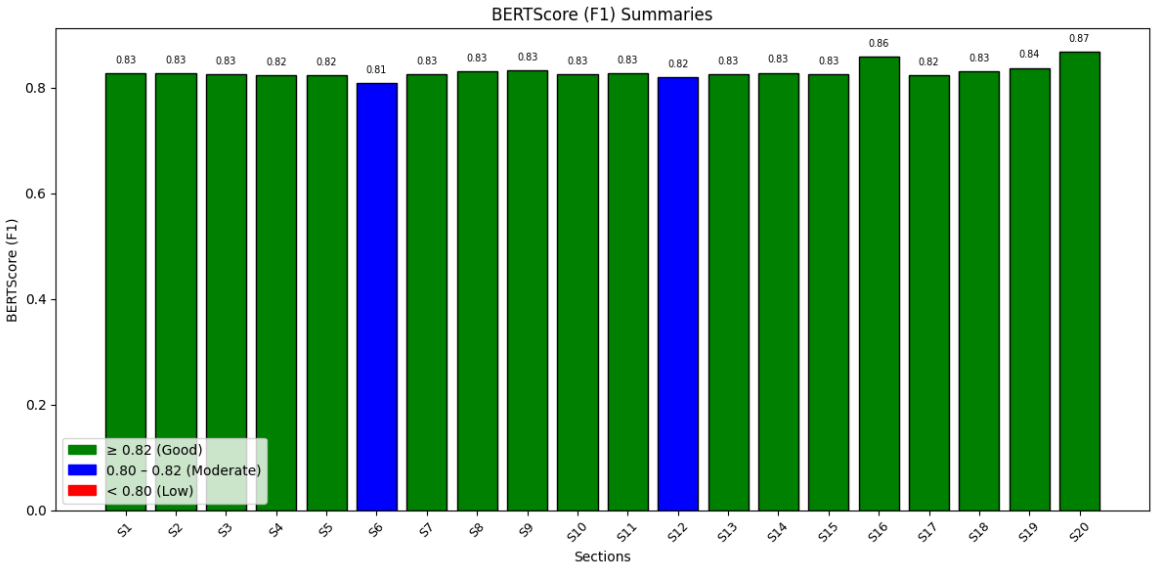

Figure 6.3: The BERTScore (F1) for the summaries of the 20 sections

As shown, the majority of sections achieve very satisfactory scores (≥ 0.83), with the highest-performing ones being §16 and §20 (BERTScore: 0.86 and 0.87, respectively). Only two sections (§6 and §12) show slightly lower scores, at 0.81 and 0.82, respectively—possibly due to greater thematic variety or higher information density in those chapters.

Table 6.3: Average Evaluation Scores for Section Summaries

Metric	**Average Score**
`BERTScore (F1)`	0.8301
`ROUGE-1`	0.0635
`ROUGE-L`	0.0583

The ROUGE metric scores are relatively low, which is attributed to the model's strong paraphrasing ability. More specifically:

- `ROUGE-1` captures the overlap of individual words between the summary and the original text.
- `ROUGE-L` measures the longest common subsequence of words.

The low ROUGE scores suggest that the model often rephrases content rather than reproducing it word for word. This, combined with the high `BERTScore` value, indicates that semantic fidelity is preserved even when different vocabulary is used.

6.5 Evaluation of Generated Question–Answer Pairs

The evaluation of the generated question–answer pairs was carried out across two main dimensions:

- **Semantic alignment between question and summary:** BERTScore (F1) was computed between each question and the summary of the section it originated from, in order to assess how well the question conceptually aligns with the source content

- **Answer correctness:** For each question, the accuracy of the corresponding answer was evaluated using an LLM via binary classification (“Yes” or “No”), based on whether the answer sufficiently addressed the question.

A total of 40 questions were evaluated (2 for each of the 20 sections). The evaluation process yielded the following indicators:

Table 6.4: Summary of Q&A Evaluation Results

Indicator	**Value**
Number of Questions	40
Correct Answers Rate	100%
Q&A Completeness	100%
Average BERTScore (F1)	0.8403

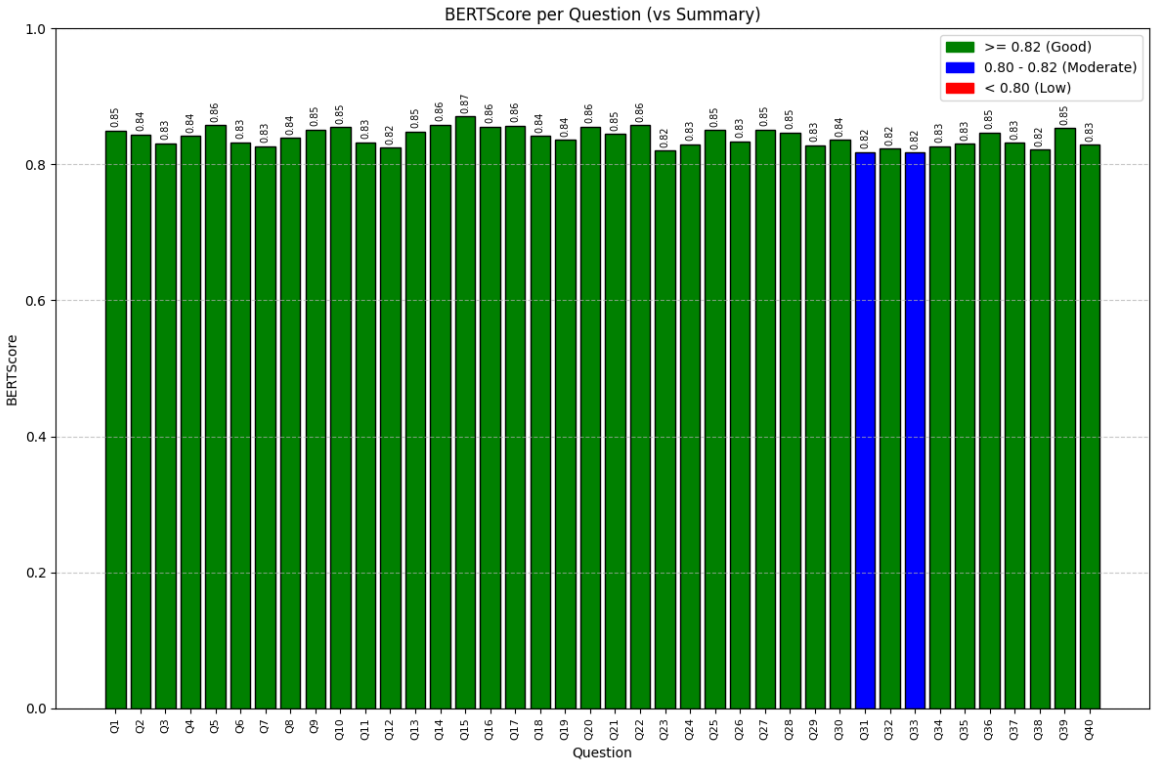

Figure 6.4: BERTScore (F1) per question - comparison with summary

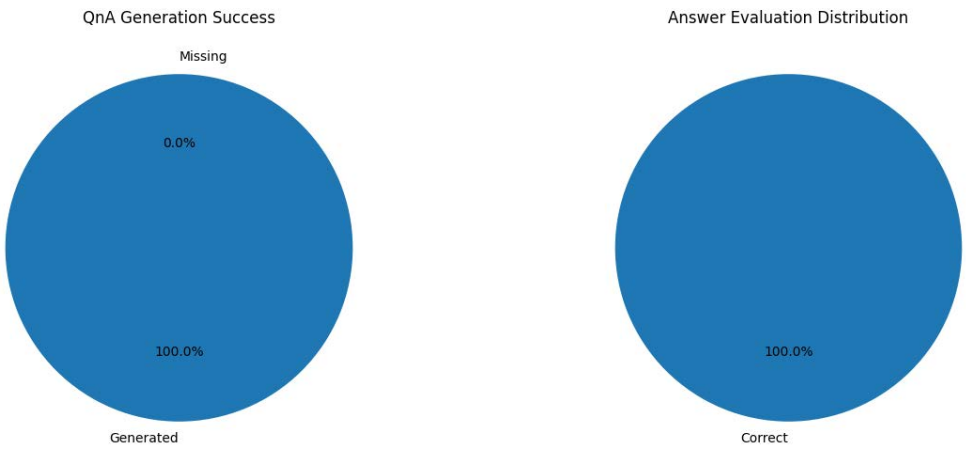

Figure 6.5: Q&A generation success rate (left) and answer accuracy (right)

As shown in the diagrams above (Fig. 6.4 and Fig. 6.5), the results demonstrate high functional accuracy and semantic validity of the generated Q&A pairs. In particular, the fact that all answers were evaluated as correct confirms the usefulness of the subsystem for reinforcement and self-assessment purposes.

6.5.1 Computational Performance

The generation and evaluation of the Q&A pairs were carried out with very low computational requirements. This is primarily due to the fact that the section summaries—on which the Q&A generation was based—had already been produced and stored during an earlier stage of the pipeline. As a result, this phase relied on preexisting summaries to generate questions and answers, without needing to reprocess the full text.

The procedure was executed in the same computational environment described in the previous chapter. Total execution time was approximately 14 minutes and memory usage remained extremely low (1.09 MB).

These results indicate that the Q&A generation and evaluation phase can be easily integrated even into resource-constrained environments, provided that intermediate data (such as summaries) are already available.

Chapter 7

Conclusions

7.1 Summary and Conclusions

This thesis explored the effectiveness of generative AI in automating services related to book indexing and associated support tasks such as marginalia generation, summarization and question–answer creation. A particularly interesting aspect of this study was that the input text was written in Greek and contained legal terminology—an element that posed a significant challenge. Through the design and implementation of a custom information system, it was demonstrated that artificial intelligence can meaningfully contribute simplifying and accelerating processes that have traditionally relied heavily on human effort.

The system evaluation showed that the generated index was able to identify approximately 37% of key terms (Recall), with a precision rate of about 26%. While these figures reveal limitations in the model's ability to accurately recognize domain-specific legal terms, they are considered satisfactory given the complexity of the content and the lack of pre-trained models tailored to the Greek legal domain. . In addition, the systematic processing of all paragraphs and the extraction of representative concepts per section contributed to the overall coherence and practical usefulness of the index.

Summaries were evaluated at two levels: (a) at the paragraph level (chunk summaries), for marginalia generation and (b) at the section level (section summaries), for the generation of question–answer pairs.

For paragraph-level summaries:

- **BERTScore (F1):** 91.46%

- **ROUGE-1:** 10.02%
- **ROUGE-L:** 9.95%

For section-level summaries:

- **BERTScore (F1):** 83.01%
- **ROUGE-1:** 6.35%
- **ROUGE-L:** 5.83%

The results indicate that the system maintains satisfactory semantic alignment with the original text in both types of summaries, despite relying on paraphrased or condensed language—an expected and desirable feature in the context of summarization. The generated marginal titles achieved an average **BERTScore (F1)** of 86.58%, alongside very low **ROUGE** scores (1.05%), confirming that the system is able to capture the thematic focus of each passage concisely, without word-for-word matching. The fact that title quality remains consistent across the dataset further supports the reliability of the generation mechanism. The evaluation of the generated question–answer pairs confirmed the effectiveness of the subsystem: out of 40 questions, results showed **100% completeness**, **100% answer accuracy** and an **average BERTScore (F1)** of 84.03%. The strong alignment between questions and summaries, along with fully correct answers, suggests that the system can effectively serve as a reinforcement and self-assessment tool for readers. It is also worth noting that the entire Q&A generation process required minimal computational resources (1.09 MB RAM), making it suitable for deployment in fully autonomous environments.

Overall, the system is not intended to replace human expertise, but rather to enhance it. The contribution of this work lies in showcasing how Artificial Intelligence can act as a support tool for information organization and automated content understanding—even in languages and domains considered challenging for current models. The results confirm that, when combined with well-designed pipelines and appropriate preprocessing, generative models can provide meaningful added value in areas such as legal documentation and educational technology.

7.2 Future Work

This study may serve as a foundation for future work in several directions:

- **Use of advanced LLMs:** Future implementations could leverage more modern LLMs, such as GPT or fine-tuned models specifically for the Greek language, aiming to improve the understanding of semantically dense parts and achieve more accurate term recognition.

- **Dynamic personalization:** The integration of machine learning techniques could allow the index or marginal headings to be adapted based on the user's profile. A student, a legal professional and a researcher each have different needs.

- **Further evaluation with human annotators:** The generated indexes, marginalia and Q&A pairs could be evaluated by domain experts to assess their practical usefulness beyond automated metrics. Crowdsourcing or peer-review within an academic setting could also be explored.

- **Expansion of output types:** The system could be extended to generate glossaries, mind maps, or interactive Q&A interfaces to better support the educational use of textbooks. Additionally, the creation of visual summaries or timelines could enhance the comprehension and retention of complex concepts.

- **Automated alignment with existing indexes:** Incorporating algorithms that compare and score the similarity between the generated output and a book's ground truth index could serve as a valuable feedback and model fine-tuning mechanism.

Through these directions, this work lays the groundwork for the use of LLMs in the organization and understanding of Greek academic content—offering a first but promising step forward.

Bibliography

[1] A. Gokul. Llms and ai: Understanding its reach and impact. *Preprints*, May 2023. Preprint.

[2] Ravil I. Mukhamediev et al. Review of artificial intelligence and machine learning technologies: Classification, restrictions, opportunities and challenges. *Mathematics*, 10(15):2552, 2022.

[3] Διαμαντάρας Κωνσταντίνος και Μπότσης Δημήτρης. *Μηχανική Μάθηση*. Κλειδάριθμος, 2023.

[4] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[5] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Convolutional networks for images, speech and time series. *Nature*, 521:436–444, 2015.

[6] Daniel Jurafsky and James H. Martin. *Speech and Language Processing*. Pearson, 3rd edition edition, 2023.

[7] Christopher D Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[8] McKinsey & Company. What is generative ai? McKinsey Explainers, 2023.

[9] Tom B. Brown et al. Language models are few-shot learners. In *NeurIPS*, 2020.

[10] Shervin Minaee et al. Large language models: A survey. *arXiv*, February 2024.

[11] Ashish Vaswani et al. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

[12] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2019.

[14] Alec Radford, Jeffrey Wu, Dario Amodei, Jack Clark, et al. Language models are unsupervised multitask learners. *OpenAI preprint*, 2019.

[15] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67, 2020.

[16] Jack Kaplan et al. Scaling laws for neural language models. *arXiv*, 2020. arXiv:2001.08361.

[17] Muhammad Usman Hadi, Qasem Al-Tashi, Rizwan Qureshi, Abbas Shah, Amgad Muneer, Muhammad Irfan, Anas Zafar, Muhammad Bilal Shaikh, Naveed Akhtar, Jia Wu, and Seyedali Mirjalili. A survey on large language models: Applications, challenges, limitations and practical usage. *arXiv preprint arXiv:2308.08264*, 2023.

[18] Nancy C. Mulvany. *Indexing Books*. University of Chicago Press, 1994.

[19] ChengXiang Zhai. Statistical language models for information retrieval: A critical review. *Foundations and Trends in Information Retrieval*, 2008.

[20] Dragomir Radev, Simone Teufel, Horacio Saggion, and Wai Lam. Evaluation of text summarization in a cross-lingual information retrieval framework. *Center for Language and Speech Processing*, 2002.

[21] Tianyi Zhang, Faisal Ladhak, Esin Durmus, and Percy Liang. Benchmarking large language models for news summarization. *Transactions of the Association for Computational Linguistics*, 2024.

[22] S. K. Ali. Title, preamble and marginal notes as intrinsic aids in the construction of statute. `https://ssrn.com/abstract=2210976`, 2013. Accessed: 2025-06-07.

[23] Pietro Angelici. Enhancing document review through knowledge graphs and large language models. `https://skemman.is/handle/1946/47695?locale=en`, 2023. Accessed: 2025-06-07.

[24] Dan Su, Yixin Xu, Tao Yu, Farhad Bin Siddique, Ehsan Barezi, and Pascale Fung. Caire-covid: A question answering and multi-document summarization system for covid-19 scholarly information management. *arXiv preprint arXiv:2005.03975*, 2020.

[25] Xin Li, Juncheng Jin, Yujia Zhou, Yiqun Zhang, and Yuxuan Zhu. From matching to generation: A survey on generative information retrieval. *ACM Transactions on Information Systems*, 2024.

[26] Kiran Pakhale. Large language models and information retrieval. *SSRN*, 2023.

[27] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[28] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[29] James G Mork, Antonio Jimeno-Yepes, and Alan R Aronson. The nlm medical text indexer system for indexing biomedical literature. *BioASQ@ CLEF*, 2013.

[30] Radim Řehůřek and Petr Sojka. Software framework for topic modelling with large corpora. *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, 46:45–50, 2010.

[31] Yashoteja Prabhu and Manik Varma. Fastxml: A fast, accurate and stable tree-classifier for extreme multi-label learning. *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 263–272, 2014.

[32] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, 2017.

[33] Yahoo! Research and Microsoft Research. Vowpal wabbit: A fast online learning system. *GitHub Repository*, 2023.

[34] Textract book indexing tool. `https://www.texyz.com/textract`, 2024. Accessed: 2025-05-14.

[35] Pdf index generator. `https://pdfindexgenerator.com`, 2024. Accessed: 2025-05-14.

[36] Picardy indexing software. `https://picardy-indexing.ca`, 2024. Accessed: 2025-05-14.

[37] Cindex indexing software. `https://www.anzsi.org`, 2024. Accessed: 2025-05-14.

[38] Xindy - index processor for latex. `https://en.wikipedia.org/wiki/Xindy`. Accessed: 2025-05-14.

[39] Marit Asula, Jane Makke, Linda Freienthal, Hele-Andra Kuulmets, and Raul Sirel. Automated subject indexing of estonian books using kratt ai. *arXiv preprint arXiv:2203.12998*, 2022. Accessed: 2025-05-14.

[40] Yubao Tang, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Shihao Liu, Shuaiqing Wang, Dawei Yin, and Xueqi Cheng. Generative retrieval for book search. *arXiv preprint arXiv:2501.11034*, 2025. Accessed: 2025-05-14.

[41] Osma Suominen. Annif: Diy automated subject indexing using multiple algorithms. *Liber Quarterly*, 29, 2019.

[42] Olena Medelyan. *Human-competitive automatic topic indexing*. PhD thesis, University of Waikato, Hamilton, New Zealand, 2009.

[43] Jennifer D'Souza, Sameer Sadruddin, Holger Israel, Mathias Begoin, and Diana Slawig. Semeval-2025 task 5: Llms4subjects – llm-based automated subject tagging for a national technical library's open-access catalog. *arXiv preprint arXiv:2504.07199*, 2025.

[44] Theoni Koutropoulou and Efstratios Gallopoulos. Tmg-bobi: Generating back-of-the-book indexes with the text-to-matrix-generator. In *IEEE International Conference on Text Mining and Data Analytics (TMDA)*. IEEE, 2023. Accessed from University of Thessaly Library.

[45] Małgorzata Marciniak, Agnieszka Mykowiecka, and Piotr Rychlik. Terminology/keyphrase extraction for creation of book indexes in polish. In *Linking Theory and Practice of Digital Libraries: 25th International Conference on Theory and Practice of Digital Libraries, TPDL 2021*, pages 49–54. Springer, 2021.

[46] Hussein Haidar and Fadi Zaraket. Back-of-the-book index automation for arabic documents. *arXiv preprint arXiv:2410.10286*, 2024.

[47] Hadrien Smart. *Making Books with Generative AI: Interfaces, Editorial Labor and Marginalia*. Master's thesis, Concordia University, 2023.

[48] Firstname Author and Secondname Author. Comparisons of quality, correctness and similarity between chatgpt-generated and human-written abstracts for basic research: Cross-sectional study. *Journal of Medical Internet Research*, 25(1):e51229, 2023.

[49] Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? *Proceedings of the 2021 ACM Conference on Fairness, Accountability and Transparency*, pages 610–623, 2021.

[50] Safiya Umoja Noble. *Algorithms of Oppression: How Search Engines Reinforce Racism*. NYU Press, 2018.

[51] Inioluwa Deborah Raji, Reuben Binns, et al. Closing the ai accountability gap: Defining an end-to-end framework for internal algorithmic auditing. *Proceedings of the 2020 ACM Conference on Fairness, Accountability and Transparency*, pages 33–44, 2020.

[52] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.

[53] Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, 2020.

[54] Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. Learning gender-neutral word embeddings. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4847–4853, 2018.

[55] Jason West and Linda Nguyen. Memory-augmented neural networks for long contexts. *Proceedings of the Neural Information Processing Systems Conference (NeurIPS)*, pages 1–12, 2022.

APPENDICES

Appendix A

Source Code and Outputs Access

The code created for this thesis is available at the following link:

The generated outputs (summaries, headings, Q&As, evaluation reports, etc.) can be accessed at: auto-book-index-marginal-headings-summaries-qna

Appendix B

Book Indexing – Creating a Semantic Index with LLM

Description

The system enables the creation of a **semantic index** from a book in PDF format using a Greek-specialized LLM (e.g., *Meltemi Q8*). For each paragraph in the book:

- A summary is generated using the LLM.
- 3–5 key concepts are extracted.
- An index is built linking concepts to their corresponding page numbers.

Usage Scenario

1. The user places the PDF file of the book into the project directory, e.g., `PROS-ETAIREIES.pdf`.
2. They execute the script:

```
python test_indexer.py
```

3. The system automatically generates:
    - A file `output_meltemi_q8.txt` with per-paragraph analysis.
    - A file `index.txt` containing the final semantic index.

System Setup and Execution

To run the system, two terminal windows are required:

1. **First terminal** – Start the Ollama server:

```
ollama serve
```

   This initializes the local LLM backend and prepares the selected model (e.g., Meltemi Q8).

2. **Second terminal** – Run the indexing script:

```
python test_indexer.py
```

   This performs paragraph processing, summarization and concept extraction.

The system connects to `http://localhost:11434` for all LLM requests.

Structure of the Generated Index (index.txt)

Example entries:

```
γνώση: pages 34, 148, 149
δάνειο: pages 453, 454
δήλωση:
    βούλησης: pages 171
    εναντίωσης: pages 190
δίκαιο:
    εμπορικής δημοσιότητας: pages 52
    της επιχείρησης: pages 24
    των προσωπικών εμπορικών εταιριών: pages 311
δαπάνες: pages 162
```

The system supports both flat and hierarchical indexing (main and sub-concepts), as shown above.

Technical Specifications

- Python 3.8+
- Ollama server running locally (`localhost:11434`)
- Greek LLM such as *Meltemi Q8*
- GPU support (optional for acceleration)

Execution Screenshot

The system requires two concurrent terminals: one running the Ollama server and one executing the indexing script. Below is an example of the actual runtime environment.

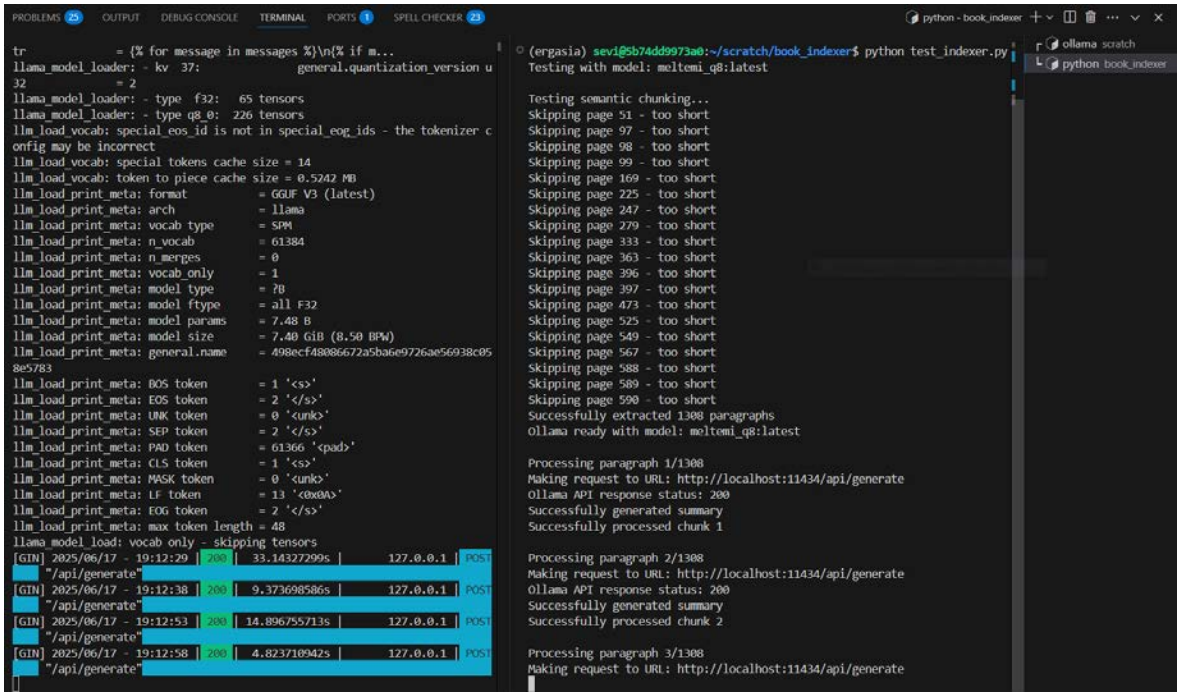

Figure B.1: Two-terminal setup during semantic chunk processing: on the left, Ollama server with API calls; on the right, the indexing script parsing and processing each paragraph.

Generated Files

After executing the indexing process, the system produces the following output files:

- `output_meltemi_q8.txt` – This file logs the LLM-generated output for each paragraph (or chunk) in the book. For every chunk, it includes:

  - The original paragraph text.
  - A concise summary of each paragraph generated by the LLM.
  - A list of extracted key concepts.

  This file provides insight into the intermediate semantic analysis performed by the model.

- `index.txt` – The final semantic index of the book. It is alphabetically structured and contains concepts (including hierarchical ones) along with the corresponding page numbers where they appear. This is the main deliverable of the indexing module.

Appendix C

Marginalia – Generating Paragraph Titles with LLM

Description

This module automatically generates **marginal headings** (short titles) for paragraphs in a Greek-language book using an LLM such as *Meltemi Q8*. For each paragraph:

- A summary is generated.
- A short, descriptive title is derived from the summary.
- Metadata such as character/word/token count is recorded.

Usage Scenario

1. The user places the input book (PDF) in the project directory with a filename such as `PROS-ETAIREIES.pdf`.
2. Then, they execute the following script:

```
python test_headings.py
```

3. The system processes each paragraph and produces:
    - One heading (title) per paragraph
    - Corresponding summary

- Evaluation and resource usage report

System Setup and Execution

As in the indexing module, two terminals are used:

1. **Terminal 1** – Start the Ollama server:

```
ollama serve
```

2. **Terminal 2** – Run the headings script:

```
python test_headings.py
```

Sample Output

The following example shows the actual output of the marginal headings module, including the raw paragraph text, the LLM-generated summary and the resulting heading.

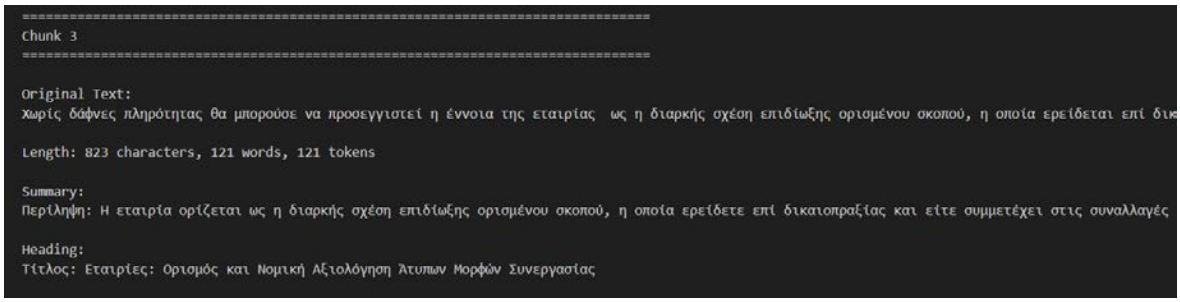

Figure C.1: Generated heading and summary for a paragraph chunk.

Technical Specifications

- Python 3.8+
- Ollama server on `localhost:11434`
- LLM such as *Meltemi Q8*
- Optional GPU acceleration

Generated Files

- `output_headings.txt` – Contains per-paragraph:
  - Original text
  - Summary
  - Marginal heading (title)
  - Character, word and token counts
  - System and performance statistics

Execution Screenshot

This screenshot shows the dual-terminal setup during heading generation. On the left, the Ollama server is running and receiving API requests; on the right, the ‘test_headings.py‘ script is processing paragraphs, summarizing them and generating marginal headings.

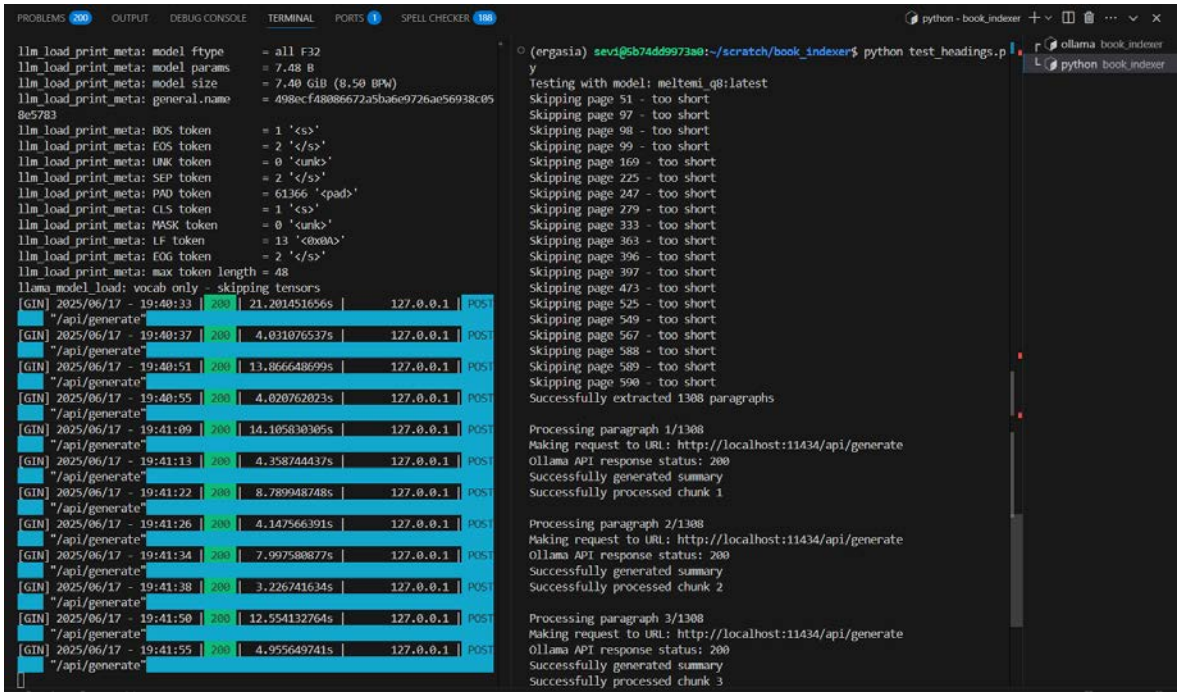

Figure C.2: Real-time heading generation using Ollama (left: server log, right: chunk processing).

Appendix D

Question and Answer Generation from Summarized Book Sections

Description

This module enables the generation of question-answer pairs for each major section (§) of a legal textbook in Greek. The system uses a local LLM (e.g., *Meltemi Q8*) to first summarize each section and then generate two meaningful Q&A pairs per summary.

The goal is to support active reading and provide automated comprehension checks.

Pipeline Overview

1. **Section Summarization**: Using `summary_per_section.py`, the text is extracted and split into chunks (max 8 pages each), then summarized via API calls to the LLM.

2. **Q&A Generation**: The script `generate_qna.py` uses the summaries to produce two Q&A pairs per section, using a templated prompt and parsing logic.

3. **Output Merging**: With `merge_summaries_with_qna.py`, the results are compiled into a structured format for review and storage.

Execution Instructions

As with previous modules, two terminals are used:

1. **Terminal 1: Start the LLM backend**

```
ollama serve
```

2. **Terminal 2: Execute the full Q&A pipeline**

```
python summary_per_section.py
python generate_qna.py
python merge_summaries_with_qna.py
```

Sample Q&A Output

Below is an example of two automatically generated question-answer pairs from section §5 and §6, displayed exactly as saved in the plain text output:

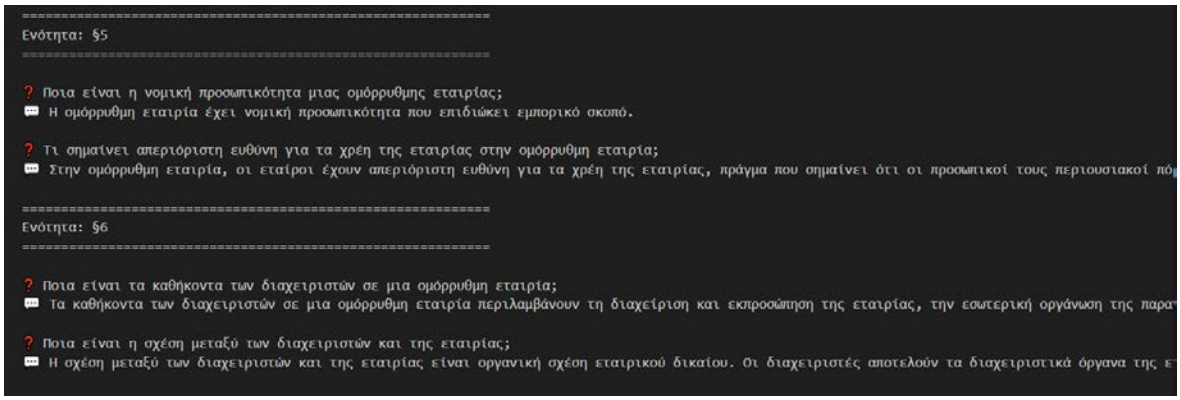

Figure D.1: Sample output from `qna_from_summaries.txt` showing two questions and answers per section.

Generated Files

- `section_summaries.csv` – One-line summary for each section (§1 to §20).
- `qna_from_summaries.txt` – Human-readable output (sections with 2 Q&A).
- `qna_from_summaries.csv` – Structured Q&A results (with raw model output).
- `qna_with_summaries.csv` – Combined summaries and Q&A per section.
- `qna_report.txt` – System metrics and duration per section.

Execution Screenshot

The following screenshots illustrate the two-step execution process for generating Q&A pairs from book sections.

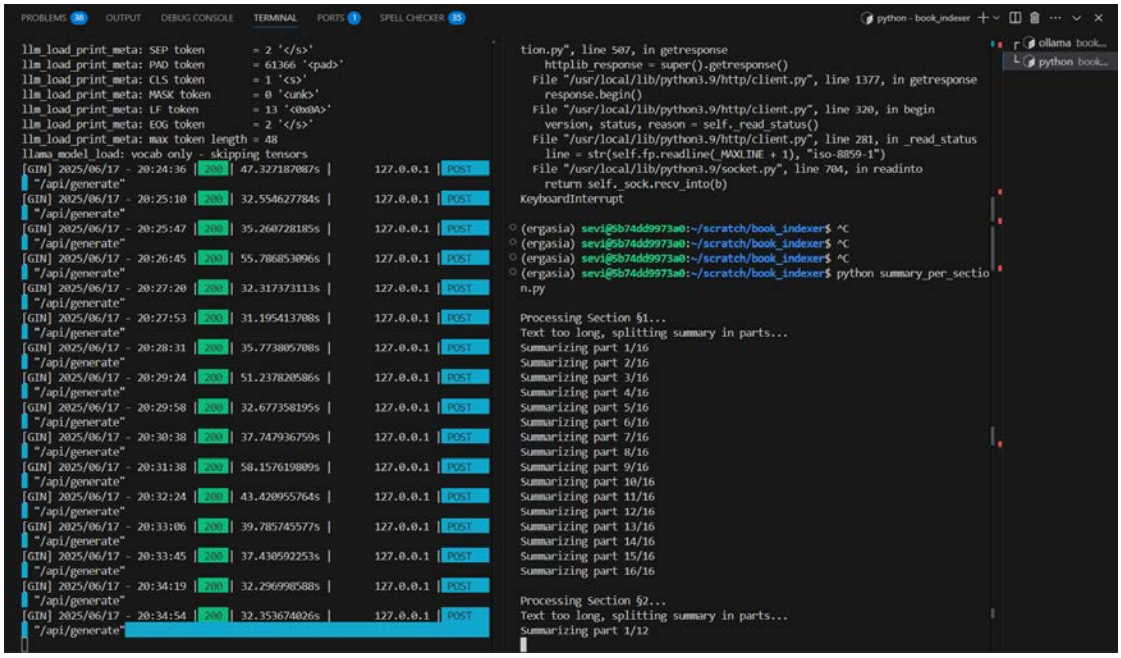

Figure D.2: Phase 1 – Section-level summarization using `summary_per_section.py`.

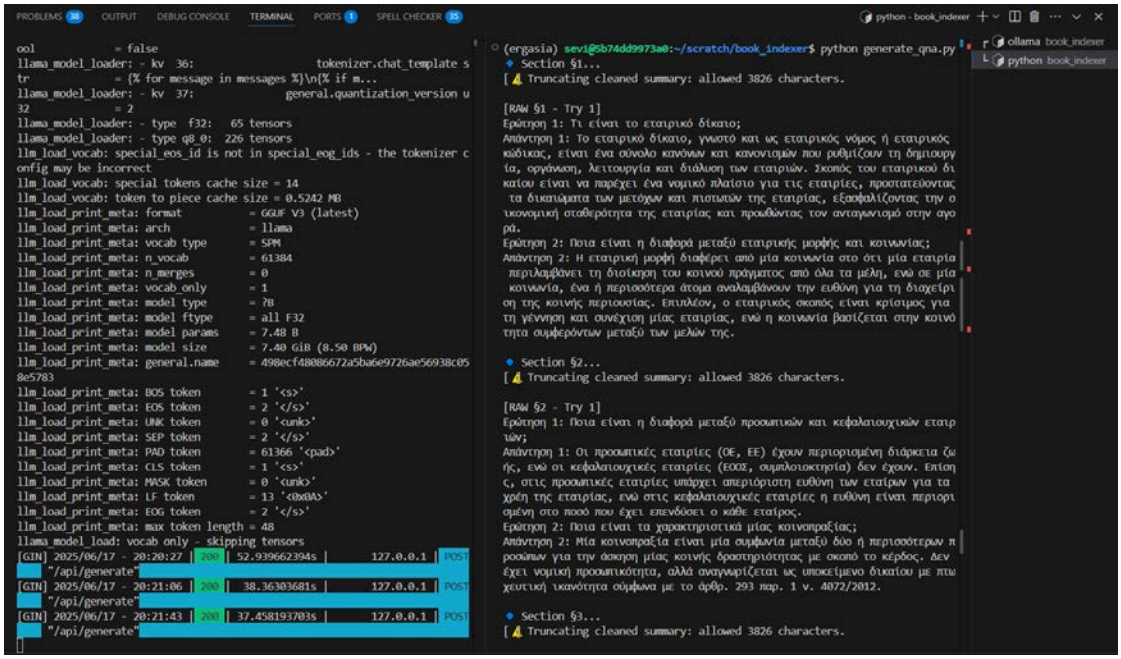

Figure D.3: Phase 2 – Q&A generation based on summaries using `generate_qna.py`.