MODULE CT6039: DISSERTATION

Enhancing Academic Rigour: A Computational Tool for Identifying Citation Gaps in Research Papers

Student Name

(Student ID: XXXXX)

This dissertation is submitted for the assessment of CT6039 at the Department of Computing. **January 5, 2026**

Abstract

Enhancing Academic Rigour: A Computational Tool for Identifying Citation Gaps in Research Papers

The credibility of academic research is fundamentally reliant on the quality and relevance of cited evidence. However, for students and researchers, manually evaluating the alignment between claims made in a text and the authority of their supporting references is a subjective and error-prone process. This dissertation presents the design and development of "RefScore," a Python-based desktop application that automates the assessment of bibliography quality and identifies citation gaps within LaTeX and PDF documents.

The research employs a software engineering methodology to construct a six-dimensional scoring algorithm that evaluates sources based on Semantic Alignment, Entity Overlap, Authority, Recency, and Methodology. The resulting artifact utilizes Natural Language Processing (NLP) to parse document structures and highlight "weak sentences"—claims that lack sufficient evidentiary support. Testing demonstrates that the tool effectively parses multi-format references (BibTeX, JSON, CSV) and provides actionable visual feedback through a PyQt5 graphical interface. This project contributes to the field of educational technology by providing a reproducible, metric-based framework for enhancing academic rigour, enabling users to proactively identify and remediate weaknesses in their research referencing.

Ethical Issues Statement

Mandatory Statement of Ethics

This dissertation project involves the design, development, and evaluation of a software artifact (RefScore) and the analysis of secondary data (academic literature).

- **Human Participants:** No human participants were involved in the primary data collection or algorithm testing phases documented in this dissertation. The system was evaluated using publicly available datasets and synthetic test documents.
- **Data Handling and Privacy:** The research utilizes only open-access academic papers and metadata (e.g., from BibTeX files and DOI records). No personal, sensitive, or confidential data was collected, stored, or processed.
- **Intellectual Property and Licensing:** All software dependencies and third-party libraries utilized in the development of the application are open-source and compliant with their respective licenses (e.g., MIT, Apache 2.0). The source code generated for this dissertation is original work and is provided in the supplementary material.

This project adheres to the university's ethical guidelines for research involving software development and secondary data analysis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

The integrity of scholarly communication is predicated on the robustness of its evidence base. In the academic ecosystem, citations serve not merely as pointers to prior work but as the epistemological foundation upon which new knowledge is constructed. Each citation represents a claim to authority, a validation of methodology, or a lineage of ideas. However, the exponential growth of scientific literature—often termed the "information explosion"—has rendered the manual verification of these foundations increasingly untenable for students and assessors alike.

For undergraduate and postgraduate students, the process of selecting high-quality sources is fraught with ambiguity. The proliferation of predatory journals, the subtle distinction between primary and secondary sources, and the difficulty of assessing ”authority” without deep domain expertise often lead to weak bibliographies that undermine the validity of their research. Furthermore, the pressure to produce content rapidly often leads to ”citation padding,” where sources are included to meet a quota rather than to genuinely support an argument.

This dissertation introduces **RefScore**, a computational solution designed to address these challenges. By implementing a multi-dimensional analysis framework within a Python-based desktop application, RefScore seeks to bridge the gap between subjective human evaluation and objective bibliographic metrics. Unlike traditional reference managers that focus on storage, or plagiarism checkers that focus on originality, RefScore focuses on *alignment* and *quality*, providing a novel tool for enhancing academic rigour.

1.2 Problem Statement

The central problem addressed by this research is the *syntactic-semantic gap* in existing reference management technologies.

Currently, the academic workflow relies heavily on Reference Management Software (RMS) such as Zotero, Mendeley, and EndNote. While these tools excel at the *syntactic* level—formatting citations according to specific styles (e.g., APA, Harvard) and organizing metadata—they fail to address the *semantic* quality of the references. They perform robust CRUD (Create, Read, Update, Delete) operations on bibliography data but offer no analytic insight into the content. They do not answer the critical questions: Does this source actually support the claim being made? Is this source authoritative within its field? Is the methodology of the cited paper aligned with the student's research?

This technological deficit creates distinct pedagogical and assessment issues:

- **The Feedback Latency:** Students often receive feedback on the quality of their sources only after summative assessment, by which time it is too late to remediate gaps in their literature review.
- **Subjectivity in Assessment:** For markers, verifying the relevance and authority of every cited source in a large cohort is practically impossible. Consequently, assessment often relies on heuristics (e.g., recognizing famous authors) rather than a systematic evaluation of evidence quality.
- **The ”Digital Hoarding” Phenomenon:** The ease of ”one-click” citation capture in modern browsers encourages students to amass large libraries of unread papers, creating a false sense of mastery without engagement with the semantic content.

There is a clear absence of tools that can ingest standard bibliographic formats (BibTeX, JSON) and provide a quantitative, multi-dimensional “score” to guide academic writing standards before submission.

1.3 Research Questions

To address the identified technological and pedagogical gaps, this project investigates the feasibility of automating citation quality analysis. The primary research question is:

- **RQ1:** To what extent can the evaluation of academic reference quality be automated using a hybrid approach of metadata analysis and Natural Language Processing (NLP)?

This overarching inquiry is supported by three sub-questions designed to guide the technical implementation and evaluation:

- **RQ2:** How can distinct bibliometric indicators—specifically "Authority," "Recency," and "Semantic Alignment"—be weighted and synthesized to produce a meaningful composite score for a reference?
- **RQ3:** How effective is the proposed algorithm in identifying "citation gaps" (weakly supported claims) within unstructured document formats such as LaTeX and PDF?

1.4 Research Objectives

The aim of this dissertation is to produce a functional software artifact that demonstrates the viability of algorithmic reference scoring. The specific objectives are:

1. **RO1:** To critically analyse existing metrics for academic source evaluation and identify suitable NLP techniques (e.g., TF-IDF, embeddings, Cosine Similarity) for semantic alignment.
2. **RO2:** To design and implement **RefScore**, a Python-based desktop application capable of multi-format ingestion (.bib, .json, .csv) and robust document parsing (.tex, .pdf).
3. **RO3:** To develop and calibrate a six-dimensional scoring algorithm that evaluates sources based on Alignment, Numbers/Units, Entities, Methods, Recency, and Authority.
4. **RO4:** To evaluate the system's efficacy by processing a test corpus of academic documents and visualising the "weakest sentences" to demonstrate utility for the end-user.

1.5 Scope and Limitations

The scope of this dissertation is defined by the development of the RefScore software and the theoretical framework underpinning its scoring logic.

- **Technical Scope:** The artifact is developed using Python 3.8+ and the PyQt5 framework for the Graphical User Interface (GUI). It prioritises local processing to ensure data privacy, utilizing libraries such as `scikit-learn` for text analysis and `bibtexparser` for metadata handling.
- **Functional Scope:** The analysis is limited to the six identified scoring dimensions. The ”Semantic Alignment” feature utilizes statistical similarity measures (TF-IDF/Jaccard) and, where available, basic transformer embeddings. It does not claim ”understanding” in the Artificial General Intelligence sense, but rather statistical correlation.
- **Linguistic Scope:** The current iteration of RefScore is optimized for English-language academic texts. Cross-lingual citation analysis is outside the scope of this project.
- **Role of Automation:** It is explicitly noted that RefScore is intended to *augment* human judgment, not replace it. The tool provides indicators of quality, but the final determination of a source’s suitability remains the responsibility of the researcher.

1.6 Dissertation Structure

Following this introduction, **Chapter 2 (Literature Review)** examines the current state of the art in bibliometrics and automated essay scoring. **Chapter 3 (Methodology)** details the software engineering principles and the design science research approach used to construct RefScore. **Chapter 4 (Results and Evaluation)** presents the implementation of the artifact and analyzes its performance against test cases. Finally, **Chapter 5 (Conclusion)** synthesizes the findings and discusses the implications for academic practice.

Chapter 2

Literature Review

2.1 Introduction

This chapter situates the development of **RefScore** within the broader academic discourse surrounding information retrieval, automated assessment, and bibliometric analysis. The assessment of academic writing quality is a multi-faceted challenge that encompasses syntactic correctness, semantic coherence, and evidentiary support. While mature technologies exist to address grammar (e.g., Grammarly) and plagiarism (e.g., Turnitin), the automated evaluation of evidentiary support remains a nascent field. This review critically analyzes existing frameworks—ranging from classical Information Behavior theory to modern Natural Language Processing (NLP)—to identify the "syntactic-semantic gap" that motivates this research.

The review is structured into four sections: Section 2.2 establishes the theoretical framework of Information Behavior, explaining *why* students struggle with source selection. Section 2.3 critiques current Reference Management Software (RMS) and Plagiarism Detection Systems (PDS). Section 2.4 explores the computational methods (NLP and AES) that underpin the proposed solution, specifically Vector Space Models. Finally, Section 2.5 presents a comparative analysis of existing tools to delineate the specific research gap RefScore intends to fill.

2.2 Theoretical Framework: Information Behavior and Citation

2.2.1 The Cognitive Burden of Source Selection

To understand the problem of poor citation quality, one must first examine the information behavior of the student researcher. The digital age has precipitated a crisis of ”Information Overload,” where the sheer volume of available literature exceeds the cognitive processing capacity of the individual. In this context, Herbert Simon’s theory of *bounded rationality* becomes critical. Simon argued that when faced with complex decisions and limited time, individuals do not optimize; they *satisfice*—choosing the first option that meets a minimum acceptability threshold.

In the context of academic writing, ”satisficing” manifests as the selection of sources that are easily accessible (e.g., the first result on Google Scholar) rather than those that are most authoritative or relevant. This aligns with Zipf’s *Principle of Least Effort*, suggesting that students will naturally gravitate towards the path of least resistance in information seeking. RefScore attempts to disrupt this behavior by providing a rapid, algorithmic ”quality check” that reduces the cognitive load of evaluating source authority, effectively lowering the barrier to rigorous selection.

2.2.2 The Role of Citation: Persuasion vs. Attribution

Citations perform two distinct functions in academic text: *attribution* and *persuasion*.

- **Attribution:** Giving credit to original authors to avoid plagiarism.
- **Persuasion:** Using the authority of a prior work to support a new claim.

Current institutional tools focus almost entirely on attribution. Plagiarism checkers verify that a source is *credited*, but they remain agnostic to whether the source is *credible*. A student may correctly attribute a statement to a non-peer-reviewed blog post, satisfying the attribution requirement, yet fail the persuasion requirement. This distinction is crucial for this dissertation, as RefScore is explicitly designed to assess the persuasive weight (Authority/Recency) and semantic relevance (Alignment) of the citation.

2.3 Technological Approaches to Academic Writing

2.3.1 The Syntactic Stagnation of Reference Management Software

Reference Management Software (RMS) has evolved significantly since the early days of index cards.

- **Generation 1.0 (Desktop):** Tools like EndNote focused on local database management and static formatting.
- **Generation 2.0 (Web/Cloud):** Tools like Zotero and Mendeley introduced cloud syncing, PDF scraping, and social discovery.

Despite these advances, modern RMS remains functionally ”syntactic.” They rely on the Citation Style Language (CSL) ecosystem to process metadata into strings (e.g., ”(Smith, 2020)”). However, they treat the bibliography as a list of independent entities. They do not analyze the *relationship* between the cited paper and the citing sentence. As noted in recent usability studies, users of Zotero often accumulate thousands of papers in their libraries (”digital hoarding”) without ever reading or evaluating them, assuming that possession of the PDF equates to knowledge of the content. This tool stagnation creates a false sense of security; the student believes that because the *format* is correct, the *reference* is valid.

2.3.2 Plagiarism Detection: The Presumption of Guilt

Turnitin is the industry standard for academic integrity. Its core algorithm utilizes ”document fingerprinting”—breaking a text into n-grams (sequences of n words) and hashing them using the Rabin-Karp algorithm or Winnowing to find matches in a database. While highly effective at detecting copy-paste plagiarism, this approach has significant limitations for quality assessment.

1. **Text vs. Meaning:** Turnitin matches text strings, not concepts. It cannot determine if a paraphrased section accurately reflects the source material.

2. **Adversarial Design:** The tool is designed to catch cheaters, not to help learners. It provides a ”similarity score” (inverse originality), which students often confuse with a quality metric. A 0% similarity score does not imply a high-quality paper; it may simply mean the student has invented false claims without citing anything.

2.4 Computational Foundations: NLP and Vector Space Models

RefScore proposes to move beyond syntax and fingerprinting by utilizing Natural Language Processing (NLP). The theoretical basis for this lies in **Vector Space Models (VSM)**.

2.4.1 Representing Meaning as Vectors

To determine if a source ”aligns” with a student’s claim, the software must compute semantic similarity. In VSM, documents are represented as vectors in a high-dimensional space.

$$d_j = (w_{1,j}, w_{2,j}, \dots, w_{t,j})$$

Where d_j is the document vector and $w_{t,j}$ represents the weight of term t.

2.4.2 TF-IDF Weighting

RefScore utilizes Term Frequency-Inverse Document Frequency (TF-IDF) to weigh the importance of words. This is critical for academic text, where common words (stopwords) like ”the” or ”and” carry little semantic value, while specific domain terms (e.g., ”algorithm,” ”biopsy”) carry high value. The weight is calculated as:

$$w_{t,d} = \text{tf}_{t,d} \times \log\left(\frac{N}{\text{df}_t}\right)$$

Where:

- $\text{tf}_{t,d}$ is the frequency of term t in document d.
- N is the total number of documents in the corpus.

- df_t is the number of documents containing term t.

The intuition behind the logarithmic scale in the inverse document frequency term is to dampen the effect of high-frequency words. Without the log, a word appearing in 1 document out of 1000 would have a weight 1000 times higher than a word appearing in all documents. The log compression ensures that rare terms are highlighted without dominating the vector completely, allowing for a balanced representation of the text's topic.

2.4.3 Cosine Similarity for Alignment Detection

Once the student's sentence (Vector A) and the cited source (Vector B) are converted into weighted vectors, the "Semantic Alignment" score is calculated using Cosine Similarity. This measures the cosine of the angle between the two vectors, providing a value between 0 (no alignment/orthogonal) and 1 (perfect alignment/parallel).

$$\text{similarity}(A, B) = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

This mathematical foundation differentiates RefScore from simple keyword matching. It allows the system to detect alignment even if the student uses slightly different phrasing, provided the vector direction (semantic theme) remains consistent. While recent advances in transformer models (e.g., BERT, OpenAI embeddings) offer context-aware embeddings, TF-IDF remains a computationally efficient baseline for local desktop applications where GPU resources may be scarce. RefScore implements a hybrid approach, preferring embeddings where available but falling back to TF-IDF.

2.5 Bibliometrics: Quantifying Authority

While NLP handles the "content," Bibliometrics handles the "context" (Authority).

2.5.1 The Challenge of Metrics

Scientometrics attempts to quantify the impact of research.

- **Journal Impact Factor (JIF):** A measure of the frequency with which the average article in a journal has been cited. While widely used, it is a controversial proxy for individual paper quality.
- **H-Index:** A metric for author productivity and impact.

Undergraduate students rarely have access to this data in a consolidated form. They cannot easily look up the H-index of every author they cite. RefScore addresses this by integrating metadata scoring.

2.5.2 The Matthew Effect

A critical consideration in automated scoring is the ”Matthew Effect” (Merton, 1968), often summarized as ”the rich get richer.” Papers that are already highly cited tend to accrue more citations regardless of their intrinsic quality, simply due to visibility. RefScore’s ”Authority” dimension attempts to mitigate this bias by considering ”Recency.” A paper from 2024 with 5 citations may be more ”impactful” than a paper from 1990 with 50 citations, relative to the time it has been available. The algorithm implements a decay function to normalize citation counts against publication age, providing a fairer assessment of contemporary literature.

2.6 Comparative Analysis of Existing Solutions

To explicitly contextualize RefScore within the current landscape, Table 2.1 critically compares the proposed system against market-leading tools across four key dimensions: Functionality, Analysis Depth, Metric Types, and User Feedback.

2.6.1 The Gap: Formative Assessment of Evidence

The comparison reveals a tripartite ecosystem:

1. **Storage Tools (Zotero):** Focus on Syntax.
2. **Policing Tools (Turnitin):** Focus on Integrity.
3. **Discovery Tools (Google Scholar):** Focus on Retrieval.

Table 2.1: Comparison of RefScore against existing academic tools.

Feature	**Zotero (RMS)**	**Turnitin (Integrity)**	**RefScore (Proposed)**
Primary Function	Reference organization and formatting (Syntax).	Plagiarism detection and text similarity.	**Reference quality** and semantic alignment.
Analysis Type	Metadata management (Title, Author, Year).	String matching (n-gram comparison).	**Multi-dimensional scoring** (NLP + Metadata).
Citation Validation	Checks style rules (APA/Harvard).	Checks existence of text in database.	Checks **semantic support** for claims.
Quality Metrics	None (User defined).	Similarity % (Inverse originality).	**Authority, Recency, Alignment, Entity Overlap.**
Gap Analysis	None.	Highlights matched text.	Highlights **unsupported claims** (Weak Sentences).
Input Format	PDF, Web, RIS.	Word, PDF.	**LaTeX (.tex), PDF, BibTeX, JSON.**

Missing from this ecosystem is a **Diagnostic Tool** focused on **Quality**. There is a clear ”Syntactic-Semantic Gap” where students are supported in *finding* papers and *formatting* citations, but unsupported in the critical intermediate step of *evaluating* whether those citations actually support their arguments.

RefScore addresses this specific gap. By synthesizing the metadata capabilities of an RMS with the Vector Space Models of AES, it proposes a novel category of tool: **Automated Citation Quality Assessment (ACQA)**. This dissertation hypothesizes that providing students with visual, quantitative feedback on their reference quality (the ”RefScore”) will disrupt ”satisficing” behaviors and lead to more rigorous academic writing.

2.7 Summary

This chapter has grounded the development of RefScore in the theories of Bounded Rationality and Information Behavior, explaining *why* students cite poorly. It has technically contextualized the project within the fields of NLP and Automated Essay Scoring, justifying the use of vector-based alignment algorithms (TF-IDF/Cosine Similarity). Finally, through a comparative analysis, it has identified a distinct gap in the market for a tool that moves beyond syntax and plagiarism to address the fundamental quality of evidentiary support. The following chapter, **Research Methodology**, will detail the software engineering process used to bridge this gap.

Chapter 3

Research Methodology and Design

3.1 Introduction

This chapter delineates the methodological framework, architectural paradigms, and algorithmic specifications governing the development of the **RefScore** artifact. Given the project's objective—to design and implement a novel software tool that addresses the "Syntactic-Semantic Gap" in academic writing—the research adopts a **Design Science Research (DSR)** approach. Unlike positivist research, which seeks to observe and describe the natural world to test hypotheses, DSR is prescriptive; it seeks to change the state of the world through the creation of innovative artifacts that solve identified problems in a specific environment.

The chapter is structured to map the Hevner et al. (2004) DSR cycles to the software development lifecycle. Section 3.2 justifies the methodological choice and details the three research cycles. Section 3.3 provides a comprehensive analysis of the system architecture, specifically the adaptation of the Model-View-Controller (MVC) pattern for local desktop deployment. Section 3.4 offers a formal specification of the scoring algorithms, including the mathematical models for recency decay and semantic alignment. Finally, Section 3.5 defines a rigorous evaluation strategy designed to validate the system's functional correctness, computational performance, and user utility.

3.2 Methodological Approach: Design Science Research

The project adheres to the DSR guidelines established by Hevner et al., which conceptualise Information Systems research as a synergy between the ”Environment” (people, organizations, technology) and the ”Knowledge Base” (foundations, methodologies). This synergy is executed through three iteration cycles:

3.2.1 The Relevance Cycle

The relevance cycle bridges the context of the research environment with the design activities. In this project, the ”environment” is the academic writing workflow of undergraduate and postgraduate students. The requirements are derived directly from the problem identified in Chapter 1: the lack of formative feedback mechanisms for citation quality.

- **Problem Definition:** Students utilize ”satisficing” heuristics to select sources due to information overload.
- **Acceptance Criteria:** The artifact must provide actionable, quantitative metrics that reduce the cognitive load of source evaluation.

3.2.2 The Design Cycle

The design cycle represents the core technical activity: the construction and refinement of the RefScore application. This project utilises an **Iterative and Incremental** development process, ensuring that critical scoring logic is validated before interface construction begins.

1. **Iteration 1: Ingestion & Parsing (Alpha).** Focus on the `refscore.core.source_loade` module. Implementation of robust parsers for BibTeX (using `bibtexparser`) and PDF text extraction (using `pdfminer.six`).
2. **Iteration 2: Algorithmic Core (Beta).** Development of the `refscore.core.scoring` engine. Integration of `scikit-learn` for TF-IDF vectorization and the implementation of the six-dimensional scoring logic.

3. **Iteration 3: Visualization & GUI (Release).** Construction of the PyQt5 frontend, integrating `matplotlib` for radar charts and establishing the signal/slot mechanism for asynchronous processing.

3.2.3 The Rigour Cycle

The rigour cycle ensures that the design process is grounded in the existing knowledge base to prevent "reinventing the wheel." RefScore draws upon:

- **Information Retrieval Theory:** Utilizing Vector Space Models (VSM) and Cosine Similarity as established methods for semantic text matching.
- **Bibliometrics:** Implementing citation half-life theories to model the "Recency" dimension.
- **Software Engineering Standards:** Adherence to PEP-8 for Python code, usage of Type Hinting for static analysis, and comprehensive unit testing via `pytest`.

3.3 System Design and Architecture

To ensure maintainability, testability, and extensibility, RefScore is architected using the **Model-View-Controller (MVC)** design pattern. This pattern decouples the internal representation of information (Models) from the user interface (View) and the algorithmic logic (Controller).

3.3.1 Architectural Justification

- **Privacy-First Local Processing:** Unlike web-based tools (e.g., Grammarly) that transmit text to remote servers, RefScore is designed as a standalone desktop application. This decision satisfies **NFR1 (Privacy)**, ensuring that users' unpublished dissertations and copyrighted PDF papers never leave their local machine.
- **Python 3.8+ Ecosystem:** Python was selected as the implementation language due to its unparalleled ecosystem for NLP and scientific computing. Libraries such as `numpy` (matrix operations) and `scikit-learn` (machine learning) provide optimized C-level performance while maintaining Python's high-level syntax.

- **PyQt5 Framework:** Selected over Tkinter for its robust widget set, support for complex event-driven programming (Signals and Slots), and native OS integration.

3.3.2 Data Flow and Processing Pipeline

The system operates through a linear processing pipeline, transforming unstructured input into structured scores:

1. **Ingestion:** The `SourceLoader` reads `.bib` files and associated `.pdf` attachments.
2. **Preprocessing:** Text is normalized (lowercased), tokenized, and filtered for stopwords (using `nltk` corpora) to reduce noise.
3. **Vectorization:** The `Analyzer` converts processed text into high-dimensional TF-IDF vectors.
4. **Computation:** The `ScoringEngine` calculates cosine similarity matrices and heuristic scores.
5. **Aggregation:** The `ScoringResult` model aggregates sub-scores using weighted sums.
6. **Visualization:** The `AnalysisTab` widget renders the results via the GUI.

3.4 Algorithmic Design Specification

The core innovation of RefScore is its ”Six-Dimensional Scoring Algorithm.” This section provides the formal mathematical specification for the key algorithms used to quantify ”Quality.”

3.4.1 Mathematical Model for Recency (S_{rec})

To model the obsolescence of scientific literature, a simple linear threshold is insufficient. Instead, RefScore implements a continuous exponential decay function, inspired by citation half-life metrics.

$$S_{rec}(t) = \begin{cases} 100 & \text{if } \Delta t \leq 3 \\ 100 \cdot e^{-\lambda(\Delta t - 3)} & \text{if } \Delta t > 3 \end{cases}$$

Where Δt is the age of the paper in years ($CurrentYear - PublicationYear$) and λ is a decay constant (default set to 0.15 to approximate a 10-year relevance half-life in Computer Science). This ensures papers published within the last 3 years receive maximum points, while older papers are penalized progressively unless manually flagged as ”Seminal.”

3.4.2 Vector Space Model for Semantic Alignment (S_{align})

As introduced in the Literature Review, alignment is calculated using Cosine Similarity over TF-IDF vectors.

$$\text{TF-IDF}(t,d) = \text{tf}(t,d) \cdot \log\left(\frac{N}{|\{d \in D : t \in d\}|}\right)$$

To optimize for performance (NFR2), the system utilizes **Sparse Matrices** (Compressed Sparse Row format). A dissertation abstract may contain unique vocabulary terms mapping to a vector space of $\mathbb{R}^{10,000+}$. Storing this as a dense matrix would consume gigabytes of RAM. Sparse representation ensures that only non-zero values are stored, allowing the application to run efficiently on consumer hardware with limited memory (8GB RAM).

3.4.3 Heuristic Scoring for Authority (S_{auth})

Determining ”Authority” without access to live API citation counts (due to the offline NFR) requires a heuristic approach based on metadata regex matching.

$$S_{auth} = (W_{venue} \cdot I_{venue}) + (W_{author} \cdot I_{author})$$

Where I_{venue} is a binary indicator function returning 1 if the publication venue matches a whitelist of prestigious journals (e.g., ”IEEE Transactions,” ”ACM Computing Surveys,” ”Nature”) and 0 otherwise. This allows the system to approximate authority based on the quality of the publication channel.

3.5 Evaluation Strategy

To ensure the artifact meets the rigorous standards of a BSc dissertation, a tripartite evaluation strategy is employed.

3.5.1 Unit Testing (Structural Verification)

The internal correctness of the code is verified using the `pytest` framework. A suite of 25+ automated tests checks edge cases, including:

- **Parser Robustness:** Capability to handle malformed BibTeX files (e.g., missing braces, non-standard encoding) without crashing.
- **Boundary Value Analysis:** Ensuring scoring algorithms return normalized values exactly between 0.0 and 100.0.
- **Mocking:** Use of `unittest.mock` to simulate PDF file reads during testing, ensuring tests are fast and independent of the file system.

3.5.2 Functional Testing (Ground Truth Validation)

The system's semantic accuracy is validated against a labeled ”Ground Truth” dataset.

- **Dataset Construction:** Two synthetic dissertations were created. Document A contains 20 references perfectly aligned with the text (High Alignment). Document B contains 20 references that are topically related but do not support the specific claims made (Low Alignment).
- **Metric:** The system's ability to assign statistically significantly higher S_{align} scores to Document A compared to Document B is tested using a T-Test ($p < 0.05$).

3.5.3 Performance Benchmarking

To verify Non-Functional Requirement 2 (Performance), the system is benchmarked on a reference machine (Intel i5, 8GB RAM). The time taken to parse and score a standard dataset (50 PDFs, 10,000 words) is recorded over 10 trials. The target threshold is set at < 60 seconds for the complete pipeline. Memory profiling (using `memory_profiler`) ensures the application does not exceed 500MB RAM usage during peak vectorization.

3.5.4 Threats to Validity

- **Internal Validity:** The heuristic nature of the ”Authority” score (regex matching) may generate false negatives for high-quality papers in niche journals not present in the whitelist.
- **External Validity:** The testing dataset is primarily English-language Computer Science papers. The tool’s efficacy in Humanities or non-English contexts remains untested.

3.6 Summary

This chapter has established the Design Science Research framework underpinning the project. It has translated the theoretical gaps identified in Chapter 2 into concrete software requirements and a robust MVC architecture. By detailing the mathematical logic of the scoring engine and the optimization strategies employed (sparse matrices), it demonstrates the technical rigour of the solution. The evaluation strategy defined here provides the roadmap for the empirical testing presented in the subsequent chapter, **Results and Evaluation**.

Chapter 4

Results and Evaluation

4.1 System Implementation

4.2 Performance Evaluation

Chapter 5

Discussion and Conclusion

5.1 Discussion of Results

5.2 Implications for Research

5.3 Future Work

Appendix A

User Manual

Appendix B

Additional Code Snippets