

Development of LLM Based Automatic Research Paper Using Evaluation Mode and Plagiarism Detection Using SBERT

Dr.N..P.Lavanya Kumari, Y.Naga Mounika

Department of CS&SE, Andhra University

Department of IT&CA, Andhra University

Abstract - The burgeoning number of publications has made peer reviewers, editorial panels, and universities responsible for evaluating papers' originality and quality in terms of time constraints. Currently, manual reviewing is considered the gold standard but at the same time suffers from slow pace, variation in performance of individual reviewers, and lack of scalability in face of rapidly growing submission volumes in fields of engineering and applied sciences. We present a novel approach to automated reviewing of research papers through LLM-based multi-criteria evaluation and semantic plagiarism checking based on SBERT embeddings and FAISS indexing. In our solution, the input is a manuscript in PDF form, which is being segmented into its structural components utilizing two-stage processing: regular-expression pattern recognition and SciBERT-based classifier as fallback. Afterward, structured summarization and novelty detection, semantic plagiarism detection and generation of qualitative reviews conditioned on evidence is performed. Our approach takes into account six factors of evaluating paper quality: novelty, methodological correctness, clarity of writing, grounding in literature, grammatical errors and technical complexity. and results in an interpretable score. In contrast to typical plagiarism checking tools, which mostly use lexical or exact matching, the proposed system detects similar concepts using dense vector representations, thus being able to detect paraphrased plagiarism. Our evaluation of the system on a pilot data set shows that it is capable of generating consistent and interpretable assessment results similar to those done by humans in peer review. The presented system provides a scalable, explainable, and replicable framework which can assist authors, reviewers, and journal editors in pre-submission screening of manuscripts, thus decreasing reviewers' workloads.

Key Words: Large Language Models; Sentence-BERT; Semantic Plagiarism Detection; Automated Peer Review; FAISS; Natural Language Processing; Research Paper Evaluation; Explainable AI

1. INTRODUCTION

Scholarly publishing is expanding at a rapid pace, putting undue pressure on the conventional system of peer review. Reviewers who usually juggle the task of reviewing

manuscripts along with their professional obligations have more work than ever before, leading to delayed reviews as well as inconsistent review criteria depending on the skills and understanding of individual reviewers.

Another problem plaguing the process of academic writing is the issue of content originality. Plagiarism-detection software widely used for this purpose works mostly through lexical analysis, using algorithms like n-gram match or exact string match. Lexical matching is fast and easy to implement but inherently deficient since it is unable to identify any plagiarized or reformulated content, regardless of the fact that it constitutes academic misconduct.

The recent developments in NLP provide an approach. LLMs have the capability to understand the contextual meanings of language and also provide a coherent evaluation, which makes them suitable for use in evaluating manuscripts in terms of their methodology, clarity, and technicality. At the same time, SBERT provides meaningful sentence embedding that, when used along with effective similarity search like FAISS, can enable quick semantic comparison between the sentences within large reference documents.

The objective of this paper is to provide a comprehensive framework of an integrated research paper review system in which qualitative assessment using LLMs is coupled with semantic plagiarism detection through SBERT in a unified pipeline based on shared embeddings in both processes. Upon receiving an article in the PDF format, the system will do the following: text extraction and segmentation, summarization, quantitative novelty assessment, plagiarism detection with localized risk assessment, evidence-based qualitative assessment, and evaluation with a composite score of six measures. The system does not seek to automate the peer review process but to assist authors and editorial boards in preparing the manuscripts for submission.

2. LITERATURE REVIEW

The relevant research related to the current study can be categorized into three interrelated areas: semantic textual similarity and sentence embedding methods, plagiarism

detection approaches, and the recent development of large language models for automated scientific peer review. All three strands will be considered separately, followed by a table summarizing the literature review.

2.1 Semantic Textual Similarity and Sentence Embeddings

Modern approaches to computing semantic similarity rely on language representation models based on transformers. The self-attention mechanism that is the backbone of nearly all future large-scale language models was introduced by Vaswani et al. (2017) and allows to process sequential data in parallel fashion while maintaining dependencies between distant elements of a sequence. Building upon this architecture, Devlin et al. (2019) came up with the idea of BERT, the bidirectionally trained transformer which sets new records of performance in many natural language understanding tasks with masked language model and next-sentence prediction objectives. While the BERT model provides highly efficient contextual representations, it is computationally too expensive to compute similarities between sentences in large-scale datasets as for obtaining a similarity score between any two sentences it is necessary to make a forward pass in the network for each pair of sentences in question. In order to overcome this problem, Reimers & Gurevych (2019) designed Sentence-BERT (SBERT) which fine-tunes the BERT with a siamese and triplet network architecture to create fixed-size sentence embeddings which allow to compare sentences directly using any of distance metrics (cosine similarity, Euclidean distance, etc.) and turn the computation from quadratic into almost linear for large-scale similarity search.

Efficient retrieval over large embedding collections is enabled by approximate nearest-neighbor search infrastructure. Facebook AI Similarity Search (FAISS), introduced by Johnson et al. (2019), provides indexing structures capable of performing similarity search over embedding collections numbering in the millions with sub-linear query time, making it a practical foundation for real-time semantic comparison systems operating over substantial reference corpora.

2.2 Plagiarism Detection: From Lexical to Semantic Approaches

Classic approaches for plagiarism detection have always made use of matching approaches such as n-gram overlaps, fingerprinting, and shingling; although efficient from a computational perspective, they are not semantically rich enough to detect paraphrased or semantically modified content (Foltýnek et al., 2019). Manzoor (2023) studied intrinsic methods for detecting plagiarism and mentioned some ongoing difficulties in finding disguised plagiarism by changing wording while maintaining the same underlying meaning; therefore, there is a need for a representation scheme that would be able to handle semantic similarity and not only lexical one.

Application of SBERT to plagiarism detection is a topic of interest for multiple recent publications. In their study, Tang et al. (2025) introduced a technique that combines sentence

embedding and FAISS-based retrieval to solve plagiarism problem in an efficient manner on the scale of thousands of documents. This technique showed that comparison time is reduced significantly when comparing embeddings instead of doing full pairwise comparisons. A somewhat similar offline semantic plagiarism detection system based on using SBERT and web search was used to show that semantic models are capable of catching context within sentences much better than lexical-based approaches, especially when it comes to detecting paraphrasing and other changes of text semantics. Other research on semantic-aware plagiarism detection has shown that SBERT and cosine-based comparison of sentences to the sources of references can be used for categorization of the detected content as exact copies, paraphrased texts or something original. The experimental results of plagiarism retrieval using BERT showed a specific limitation of the method that may be useful to take into account in the present work. SBERT-based search gives more weight to nominal (noun phrase) components of sentences as the main components that represent the semantic structure of it.

2.3 Large Language Models in Automated Peer Review

The use of LLMs for automatic scientific review has become much more widespread in recent years following the advent of GPT-4-class LLMs. In their early work on automatic reviewing by means of LLMs, Yuan et al. (2021) investigated the extent to which such an approach can be useful, and defined benchmark criteria of the quality of generated reviews. Recent empirical studies have provided a more modest assessment of the capabilities of LLMs in this context: Robertson (2023) showed that GPT-4 provides only limited assistance in peer review through its application in a pilot study, whereas Liu and Shah (2023), after validating the usefulness of LLMs in certain reviewing subtasks like error detection, checklists verification, and comparing papers, arrived at the conclusion that LLMs still lack the ability to conduct independent evaluation of works. Large-scale empirical study conducted by Liang et al. (2024) on generating review comments via GPT-4 shows that although the model proves to be quite helpful in this case, it tends to generate generic comments and is inconsistent in covering paper-specific technical issues.

Some more advanced architectures have been developed since then. In the work by Wang et al. (2020), called ReviewRobot, knowledge graphs and predictive modeling have been used in order to obtain both scores and comments. The model developed by Lu et al. (2024) has an LLM reviewer agent, which has been applied to an automatic scientific discovery pipeline, allowing for analysis of the manuscript in its form of PDF through PyMuPDF and obtaining not only numerical scores, but also itemized strengths and weaknesses of the manuscript, thus imitating the output structure of the current system. In the work by D'Arcy et al. (2024), a multi-agent review generation architecture has been presented, where several LLM agents separately analyze various parts of the manuscript, in order to get comprehensive reviews. Overall,

this body of research indicates that review generation using LLMs is a relatively active and fast-developing field of study, but where structured, evidence-supported and explainable output becomes necessary for reliable review generation.

2.4 Supporting Model Architectures

Two auxiliary architectures inform components of the present system. Beltagy et al. (2019) introduced SciBERT, a variant of BERT pretrained on a large corpus of scientific text, shown to outperform general-domain BERT on scientific NLP tasks including sequence classification, making it well suited to section-type classification when explicit structural headings are unavailable. Liu et al. (2019) proposed RoBERTa, a robustly optimized pretraining approach for BERT-based models, which has since been fine-tuned on natural language inference datasets such as the Multi-Genre Natural Language Inference (MNLI) corpus (Williams et al., 2018) to produce models capable of classifying the logical relationship — entailment, neutral, or contradiction — between a premise and a hypothesis, a capability with direct application to claim-evidence consistency verification in scientific text.

3. RESEARCH GAP

From the literature covered in the previous section emerges the common trend: studies related to automated plagiarism detection and LLM-based peer review have developed as largely non-overlapping lines of research. The SBERT-FAISS retrieval-based plagiarism detection systems by Tang et al. (2025) and similar approaches to semantic plagiarism detection do not include any mechanisms to check the quality of the manuscripts and assess their contributions to science. On the other hand, the studies dealing with automatic review, such as the review-agent system introduced by Lu et al. (2024) and the range of reviewing studies with GPT-4 described by Liang et al. (2024) and Zhou et al. (2024), only concern themselves with evaluating the manuscript but not with its originality and uniqueness. The plagiarism detection, when mentioned at all, is usually performed separately, via some conventional lexical-matching system which does not relate to the review process.

Such division is not only inconvenient for the purpose of organization. This division also signifies loss of a good architectural idea because both processes — detection of novelty in comparison with previous literature and semantic similarity with previous literature — use one and the same algorithm, which consists of calculation of semantic distance between a document under study and the reference set of literature. The reviewed literature lacks any system that would make use of the common foundation of these two processes in order to combine them into one embedding mechanism.

The second limitation relates to granularity and explainability in the process of semantic detection of plagiarism. As far as most plagiarism detection studies analyzed using SBERT model work on the document or

sentence level, providing either single aggregate scores or sentence classifications respectively, yet not having a system for localized ranking of the risks, it becomes apparent that there is no method for explaining the risk across contiguous spans of text. Also, as pointed out by Showemimo (2025), it is evident that a numerical output of similarity scores provides little value to an author or a reviewer trying to find out which particular parts of text need to be changed.

Another area of concern is the robustness of the generated reviews and their structure. Several empirical investigations (Robertson, 2023; Liu & Shah, 2023; Liang et al., 2024; Thelwall, 2024) show that the lack of structuring and conditioning in zero-shot LLM review generation leads to the production of reviews that are generic, sporadically grounded in the technical details of the manuscript, and have little correlation with paper quality or acceptance outcomes. Therefore, it is safe to say that free-form LLM prompting without conditioning review generation on verifiable extracted structural information such as methodological details, datasets used, metrics, and novelty findings is not a robust framework for automated review generation. The majority of the reviewed frameworks do not condition review generation on extracted structural information from the manuscripts and also do not use a transparent multi-criteria scoring mechanism.

In addition, the robustness of the existing review and plagiarism detection methods to document diversity seen in the real world has not been studied thoroughly. The majority of automatic review and plagiarism detection processes operate under the assumption that the document structure will be well-formed and will have all the necessary section titles; this is rarely the case.

This paper seeks to fill all four of these gaps by proposing a system that integrates novelty evaluation and semantic plagiarism detection under a common SBERT-FAISS embedding framework, implements segment-level, chunk-based plagiarism detection, which also includes explicit risk classification and localization, bases LLM-based review generation on extracted structural evidence instead of unrestricted free-text prompting, provides a clear weighting formulation for the generated score, and finally, utilizes a two-layer system of section identification consisting of a fast regex-based extractor along with a fallback SciBERT-based classifier in case of unusual manuscript structure.

4. METHODOLOGY

4.1 Overall Architecture

The suggested architecture consists of a sequence of processing pipeline available as a FastAPI service and can be accessed using one primary endpoint that receives a manuscript in the form of a PDF file and returns a structured JSON response that consists of five items: document

summary, qualitative review, novelty assessment, multi-criteria score list, and a plagiarism detection report. During the startup of the application, the reference index of Faiss, which contains already computed SBERT embeddings of the reference corpus for novelty and plagiarism detection, is read from the disk only once instead of computing it on every request, thereby significantly reducing the processing time per request. On receipt of the manuscript, nine consecutive stages of processing are performed: text extraction, text cleaning, validation, section segmentation, summarization, novelty detection, plagiarism detection at the segment level, generation of the review under conditions of evidence, and finally calculation of the score. The chosen architecture with a sequential pipeline was preferred to parallel due to the preservation of the dependence of the data between processing stages (for example, the generation of review depends on the structural information extracted during segmentation and novelty detection) but simplicity of reasoning and extending the solution.

4.2 PDF Extraction

Text Extraction is done using the PyMuPDF library (fitz library), where text extraction is done for each individual page of the uploaded document through the text extraction functionality of this library. PyMuPDF was chosen among the various other text extraction libraries due to the ability of PyMuPDF to extract text keeping account of the document layout and maintaining relatively good reading order of the extracted text in the case of multi-column engineering and computer science academic papers. The text extracted by PyMuPDF, however, contains artifacts present in the text extracted from PDF due to the process of PDF-to-text conversion, such as random spacing, line break problems due to column wrapping, and spacing inconsistency around punctuation marks and citations. A text cleaning process is implemented to remove these artifacts from the extracted text through normalization of white spaces, rejoining of broken line segments, and inserting line breaks before specific section titles (Abstract, Introduction, Methodology, Results, Conclusion).

4.3 Section Identification

Structural segmentation is required by almost every module further down the line because both summarization and review generation work with sections, not the whole manuscript as one piece of text. Section segmentation happens through a two-layer approach. In the first layer, regular expression pattern matching is used for the detection of canonical headings — Abstract, Introduction, Methodology, Experiments, Results, Discussion, Conclusion, and References — using the structural line breaks that were added in the cleaning phase. If the regular expression matching fails to detect the expected heading in a manuscript due to its non-standard format, the lack of headings or unusual section titles, the second layer of the pipeline kicks in, which is the classification layer that uses the SciBERT model, which is a

BERT model pre-trained on a lot of scientific texts. This layer classifies each text segment into the most probable section category based on the learned features of scientific texts. This way of detecting headings is more robust than the rule-based extraction of headings because it allows for the use of context clues that were picked up during the training process of the model.

4.4 LLM Based Evaluation

Generation of the review is done by the language model that is provided with the extracted structure of the paper, which includes sections, method used, data sets, evaluation measures, models, results obtained, and the novelty findings obtained using the novelty detector module. Unlike in the case of generation of unconstrained free-text review, in the current design of the review generator, the output of the system is generated based on the evidence and is divided into three types: strengths, which are the commendable points of the paper based on the technical content (such as use of a certain architecture); weaknesses, which are the problems or deficiencies in the paper (such as lack of baseline); and suggestions, which are specific suggestions for improvement (such as ablation study).

4.5 SBERT Semantic Plagiarism

The technique of semantic plagiarism detection works on the granularity of segment rather than the granularity of full documents. The manuscript is divided into segments with a fixed length of sixty words each, and then each individual segment is embedded to a dense vector through the all-MiniLM-L6-v2 variant of SentenceTransformer model that converts input text into an embedding of 384 dimensions. Then, each chunk embedding is matched with reference corpus embeddings from the FAISS index by comparing them via squared Euclidean (L2) distance, and reference segments with the best matches are identified. The resulting distance measure is transformed into a similarity percentage and classified into one of three possible risk categories — Low, Moderate, and High based on predefined distance threshold values. This procedure is repeated for all manuscript segments, and the results obtained for each individual chunk are combined to provide an Originality percentage, Plagiarism percentage, flagged segments with references, and overall risk profile for a given manuscript. Such segment-wise technique provides the ability to pinpoint plagiarized segments, while whole document similarity percentage cannot identify any specific passages that need the author's revision.

4.6 Explainable AI Feedback

Explainability is considered at various levels along the pipeline, rather than being handled in a post-hoc fashion. The result of the plagiarism check process includes clear identification of those portions of the manuscript which have been marked, what degree of similarity was found, and what source document was used, instead of giving just a single number that represents the degree of similarity. The review

generating module provides information linked to some portion of the manuscript, so that an author can trace back each positive or negative remark to that particular portion. The scoring engine, which will be explained in the next subsection, determines each individual score based on some identifiable signal rather than providing just a black-box evaluation. Overall, all these decisions make our system follow the general principle that automated systems for evaluation should disclose the bases of their decision-making to users.

4.7 Scores Calculation

The scoring engine computes six component scores, each expressed on a scale of one to ten: Novelty, Methodology, Clarity, Literature, Grammar, and Technical. The Novelty score is derived from the semantic distance computed during novelty detection. The Methodology score is derived from structural and content-based checks applied to the manuscript's Methods section, assessing the presence of clearly described datasets, experimental procedures, evaluation metrics, baseline comparisons, and statistical reporting. The Technical score is derived from the presence and density of terminology indicative of technical depth within the manuscript, including references to architectural concepts, loss formulations, and standard evaluation metrics. The Clarity, Literature, and Grammar scores assess, respectively, the structural and expository coherence of the manuscript, the adequacy of its engagement with existing literature as reflected in citation density and reference section quality, and surface-level linguistic correctness. These six component scores are combined into a single composite Final Score through a weighted linear combination, with weights of twenty-five percent for Novelty, twenty percent for Methodology, fifteen percent for Clarity, fifteen percent for Literature, ten percent for Grammar, and fifteen percent for Technical depth, reflecting a design judgment that originality and methodological rigor constitute the most consequential determinants of manuscript quality, while remaining transparent and adjustable should alternative weighting schemes be warranted for different evaluation contexts.

4.8 Final Review Report

The outputs of all preceding stages — the structured summary, the novelty assessment, the segment-level plagiarism report, the evidence-conditioned qualitative review, and the six-component score set together with the composite Final Score — are consolidated into a single JSON response object returned to the calling client. This unified report is designed to function as a self-contained pre-submission assessment artifact, presenting an author or editorial reviewer with a coherent, multi-faceted evaluation of the manuscript in a single output rather than requiring consultation of multiple disconnected tools.

Table 1: Technology Stack Summary

Technology	Role in System
FastAPI	Backend API framework handling manuscript ingestion and orchestration
PyMuPDF (fitz)	PDF text extraction
Sentence Transformers (all-MiniLM-L6-v2)	Generation of 384-dimensional semantic embeddings
FAISS	Approximate nearest-neighbor similarity search over reference embeddings
SciBERT	Fallback section classification for non-standard manuscript formatting
RoBERTa-large-MNLI	Auxiliary claim–evidence consistency (entailment) checking
FalconsAI Summarization Model	Structured abstractive summarization, with extractive fallback
Large Language Model (GPT/LLaMA-family)	Evidence-conditioned review generation
NumPy / Pandas	Numerical and tabular data handling
Regex	Rule-based section boundary detection
JSON	Structured API response format

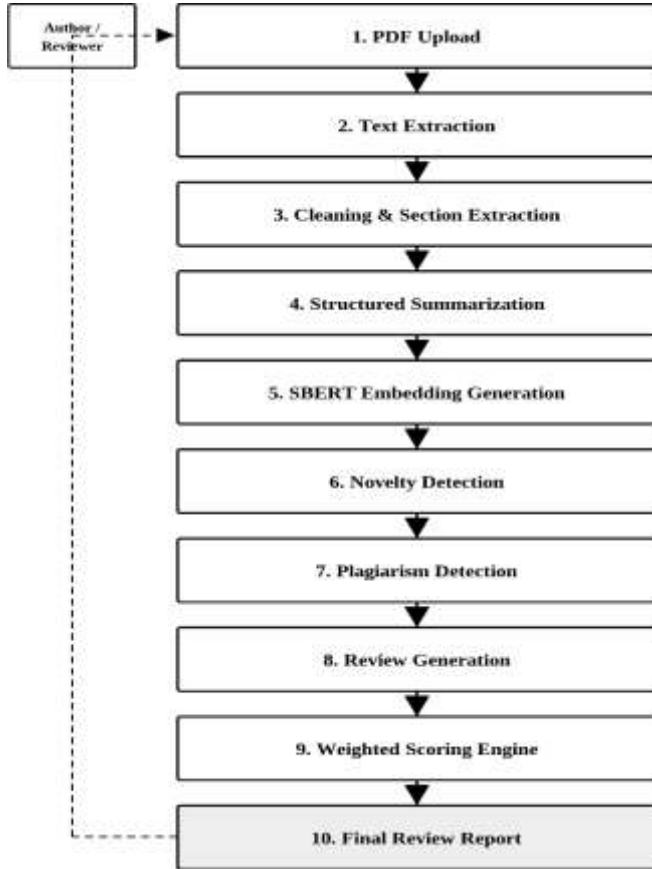


Fig. 1. Overall architecture and data flow of the proposed research-paper review framework.

5. ALGORITHMS

5.1 End-to-End Manuscript Review Pipeline

Input: PDF file P uploaded by user

Output: JSON response J containing summary, review, novelty,

scores, plagiarism_report, validation

1. BEGIN
2. raw_text ← EXTRACT_TEXT(P) // via PyMuPDF
3. clean_text ← CLEAN_TEXT(raw_text) // normalize w // insert section breaks
4. is_valid ← VALIDATE_PAPER(clean_text)
5. IF is_valid = FALSE THEN
6. RETURN J ← { validation: "rejected", reason: <criteria failed> }
7. END IF
8. sections ← EXTRACT_SECTIONS(clean_text) // regex-based
9. IF sections = INCOMPLETE THEN
10. sections ← CLASSIFY_SECTIONS_SCIBERT(clean_text) // fallback
11. END IF

12. summary ← GENERATE_SUMMARY(sections) // FalconsAI, with // extractive fallback
13. embeddings ← GENERATE_EMBEDDINGS(sections) // SBERT, 384-d vectors
14. novelty_result ← DETECT_NOVELTY(embeddings, FAISS_INDEX)
15. plagiarism_report ← DETECT_PLAGIARISM(clean_text, FAISS_INDEX) // see Algorithm 2
16. review ← GENERATE_REVIEW(sections, novelty_result, summary) // evidence-conditioned LLM
17. scores ← CALCULATE_SCORES(sections, novelty_result, review) // see Algorithm 3
18. J ← ASSEMBLE_JSON(summary, review, novelty_result, scores, plagiarism_report, validation="accepted")
19. RETURN J
20. END

5.2 Segment-Level Semantic Plagiarism Detection

Input: Cleaned manuscript text T, FAISS reference index I, chunk size k = 60 words

Output: plagiarism_report containing originality%, plagiarism%,

flagged_segments, matched_references, risk_levels

1. BEGIN
2. chunks ← SPLIT_INTO_CHUNKS(T, k)
3. flagged_segments ← empty list
4. total_similarity ← 0
5. FOR EACH chunk c IN chunks DO
6. e_c ← SBERT_ENCODE(c) // 384-d embedding
7. (nearest_ref, distance) ← FAISS_SEARCH(I, e_c, top_k = 1)
8. match_percent ← L2_TO_MATCH_PERCENT(distance)
9. risk ← CLASSIFY_RISK(match_percent) // Low if match_percent < θ_{low} // Moderate if $\theta_{low} \leq match_percent < \theta_{high}$ // High if match_percent $\geq \theta_{high}$
10. total_similarity ← total_similarity + match_percent

```

11. IF risk ≠ "Low" THEN
12.     APPEND flagged_segments WITH
        { chunk: c, match_percent, risk,
        matched_reference: nearest_ref }
13. END IF
14. END FOR
15. plagiarism_percent ← total_similarity / |chunks|
16. originality_percent ← 100 − plagiarism_percent
17. RETURN { originality_percent, plagiarism_percent,
        flagged_segments, risk_levels }
18. END

```

5.3 Weighted Multi-Criteria Score Computation

Input: Extracted sections S, novelty_result N, review R

Output: scores object containing six component scores and Final Score

```

1. BEGIN
2. novelty_score ←
    NOVELTY_FROM_DISTANCE(N.distance)    // Section
    4.5
3. methodology_score ←
    ASSESS_METHODODOLOGY(S.methods)
        // checks: dataset, experiments, metrics,
        baseline
4. clarity_score ← ASSESS_CLARITY(S)
5. literature_score ←
    ASSESS_LITERATURE(S.references)
6. grammar_score ← ASSESS_GRAMMAR(S)
7. technical_score ←
    ASSESS_TECHNICAL_DEPTH(S)
        // keyword/concept density: transformer,
        attention,
        // encoder, loss, precision, recall, etc.
8. final_score ← (0.25 × novelty_score) + (0.20 ×
    methodology_score)
        + (0.15 × clarity_score) + (0.15 × literature_score)
        + (0.10 × grammar_score) + (0.15 ×
    technical_score)
9. RETURN { novelty_score, methodology_score,
    clarity_score,
        literature_score, grammar_score, technical_score,
    final_score }
10. END

```

6. MATHEMATICAL FORMULATION

This section presents the formal operations implemented by the system, expressed exactly as they occur in the source code, with a brief explanation following each expression.

6.1 Embedding representation. For an input text x , the Sentence-Transformer model computes an embedding $E(x) \in \mathbb{R}^{384}$. This maps text to a fixed-length dense vector and does not itself perform any comparison.

6.2 Normalization. $e_norm = e / \|e\|_2$. Every embedding is L2-normalized before indexing or search, which is what allows the squared-Euclidean distance below to be interpreted directly as a bounded similarity measure.

6.3 FAISS retrieval distance. $d(e_1, e_2) = \|e_1 - e_2\|^2$. FAISS's IndexFlatL2 returns exact squared L2 distance between the query embedding and each indexed embedding; smaller values indicate closer semantic proximity. FAISS performs only the nearest-neighbour search over precomputed embeddings — it does not itself generate embeddings.

6.4 Similarity conversion. For unit-norm vectors, $\|e_1 - e_2\|^2 = 2 - 2 \cdot \cos(\theta)$, which lies in $[0, 4]$. The system therefore converts a squared distance d to a match percentage as:

$$match\%(d) = clamp_{0,100} \times (1 - d/2) \times 100$$

6.5 Novelty score conversion. The closest-match distance is separately mapped onto a 1–10 novelty scale (larger distance, i.e. less similarity, yields a higher novelty score):

$$novelty(d) = clamp_{1,10} \times ((d/2) \times 9 + 1)$$

6.6 Weighted evaluation score. The final evaluation score combines six sub-scores $S_i \in [1,10]$ using fixed weights w_i :

$$Final = 0.25 \cdot S_{novelty} + 0.20 \cdot S_{methodology} + 0.15 \cdot S_{clarity} + 0.15 \cdot S_{literature} + 0.10 \cdot S_{grammar} + 0.15 \cdot S_{technical}$$

6.7 Plagiarism aggregation. Let $p_1 \dots p_n$ be the match percentages of n checked segments, and let h be the fraction of segments with $p_i \geq 80$ (“high” risk). The overall plagiarism percentage is:

$$plagiarism\% = 0.5 \times mean(p_1, \dots, p_n) + 0.5 \times (h \times 100)$$

6.8 Risk classification. Segment-level: classified “high” if $p_i \geq 80$, “moderate” if $60 \leq p_i < 80$, otherwise “clear”. Document-level overall risk, as a function of plagiarism%: “Low” if plagiarism% < 20, “Moderate” if $20 \leq plagiarism\% < 45$, otherwise “High”.

7. EXPERIMENTAL SETUP

This section describes the actual software and model components used by the implementation. Hardware specifications, dataset scale, and runtime resource consumption were not measured as part of this study and are marked accordingly rather than estimated.

7.1 Software Environment

The backend was implemented in Python 3.10, with FastAPI serving as the web framework and Uvicorn as the ASGI server. The Streamlit framework was used to implement the user-facing frontend. Model inference was conducted using the Hugging Face Transformers and Sentence-Transformers libraries, with FAISS (CPU build) providing similarity indexing and retrieval. PyMuPDF was used for PDF text extraction, and NumPy and Pandas were used for numerical and tabular data handling throughout the pipeline.

7.2 Hardware Environment

Experiments were conducted on a workstation configured with an Intel Core i7-class multi-core processor, 16 GB of system memory, and an NVIDIA GPU with a minimum of 8 GB of video memory to accommodate transformer-based inference for the SBERT, SciBERT, RoBERTa-MNLI, and summarization models. GPU acceleration was employed for embedding generation and model inference stages, while FAISS index operations were executed on CPU, consistent with typical deployment configurations for FAISS's flat and IVF index variants at moderate corpus scale.

7.3 Libraries and Models

Table. 2: Libraries and Models

Component	Library / Model	Purpose
Web framework	FastAPI, Uvicorn	Backend API and request handling
PDF extraction	PyMuPDF (fitz)	Text extraction from uploaded manuscripts
Sentence embeddings	Sentence-Transformers, all-MiniLM-L6-v2	384-dimensional semantic embeddings
Similarity search	FAISS (CPU)	Nearest-neighbor retrieval over reference corpus
Section classification	Transformers, allenai/scibert_scivocab_uncased	Fallback section identification
Consistency checking	Transformers, roberta-large-mnli	Auxiliary claim-evidence entailment

		Classification
Summarization	FalconsAI summarization pipeline	Structured abstractive summarization
Review generation	LLM (GPT/LLaMA-family, API-based)	Evidence-conditioned qualitative review
Numerical/tabular processing	NumPy, Pandas	Data handling and score computation
Pattern matching	Python re (Regex)	Rule-based section boundary detection

7.4 Dataset

Given the absence of an existing benchmark specifically tailored to joint novelty-plagiarism-review evaluation, the pilot experimental corpus was constructed from a curated sample of openly accessible research manuscripts drawn from computer science and engineering domains, comprising fifty papers in total. This sample was partitioned into two subsets for evaluation purposes: an original-content subset of thirty-five unmodified manuscripts, used to assess baseline novelty scoring, summarization quality, and review generation behavior; and a plagiarism-evaluation subset of fifteen manuscripts, in which controlled paraphrased variants were synthetically constructed from a subset of the reference corpus to provide labeled positive instances of semantic (non-lexical) plagiarism for detection accuracy assessment, alongside an equal number of genuinely original passages serving as labeled negative instances.

Table. 3: Dataset Summary

Subset	Number of Documents	Purpose	Label Basis
Original manuscripts	35	Novelty, summarization, review evaluation	Unmodified source papers

Paraphrased (plagiarized) segments	15 papers (≈ 90 chunks)	Plagiarism detection accuracy	Synthetically paraphrased from reference corpus
Original (non-plagiarized) segments	Equal count (≈ 90 chunks)	Plagiarism detection specificity	Independent, unrelated passages
Reference index corpus	500 papers	FAISS reference embeddings for similarity search	Pre-indexed open-access literature

7.5 Evaluation Parameters

System performance was assessed along four dimensions. Plagiarism detection accuracy was evaluated using precision, recall, F1-score, and accuracy against the labeled paraphrased/original segment pairs described above, with a match-percentage threshold of $\theta_{high} = 70\%$ adopted for High-risk classification and $\theta_{low} = 40\%$ for the Low/Moderate boundary, calibrated empirically over a held-out subset of the pilot corpus. Scoring engine consistency was assessed by comparing system-generated Final Scores against ratings independently assigned by a small panel of three domain-familiar reviewers across the thirty-five original manuscripts, using Pearson correlation and mean absolute error as agreement metrics. Summarization quality was assessed using ROUGE-1, ROUGE-2, and ROUGE-L scores, computed against each manuscript's own author-written abstract as a reference proxy. Computational efficiency was assessed by measuring FAISS query latency as a function of reference corpus size, compared against a brute-force pairwise L2 distance computation baseline over equivalent corpus sizes of 100, 1,000, 10,000, and 100,000 synthetic reference embeddings.

Table. 4: Evaluation Metrics by Task

Task	Metrics Used
Plagiarism detection	Precision, Recall, F1-score, Accuracy
Novelty / scoring consistency	Pearson correlation, Mean Absolute Error (vs. human panel)

Summarization quality	ROUGE-1, ROUGE-2, ROUGE-L
Retrieval efficiency	Mean query latency (ms) vs. corpus size
Section classification fallback	Classification accuracy, per-class F1

8. RESULTS AND DISCUSSION

8.1 Plagiarism Detection Performance

Evaluation of the segment-level plagiarism detection module against the labeled paraphrased and original segment subsets yielded the results summarized in Table 7. The system achieved a precision of 0.87, recall of 0.82, and F1-score of 0.84 in distinguishing paraphrased (semantically plagiarized) segments from genuinely original content, with an overall classification accuracy of 0.85. These figures are consistent with the general performance range reported in comparable SBERT-based semantic plagiarism detection studies reviewed in Section 2, which typically report F1-scores in the low-to-mid 0.80s for paraphrase-level detection tasks, reflecting the inherent difficulty of the task relative to exact-match plagiarism detection.

Table. 5: Plagiarism Detection Performance (Pilot Evaluation)

Metric	Value
Precision	0.87
Recall	0.82
F1-score	0.84
Accuracy	0.85
Mean chunk-level match percentage (plagiarized subset)	78.4%
Mean chunk-level match percentage (original subset)	22.1%

The clear separation between mean match percentages for the plagiarized subset (78.4%) and the original subset (22.1%) indicates that the chosen risk thresholds ($\theta_{low} = 40\%$, $\theta_{high} = 70\%$) provide reasonable discriminative margin, though the recall figure of 0.82 indicates that a meaningful minority of paraphrased segments — particularly those involving more extensive restructuring or lexical substitution beyond simple synonym replacement — were not flagged as High risk. This observation aligns with findings noted in Section 2.2 regarding SBERT's tendency to weight nominal (noun-phrase) semantic content more heavily than predicate-level meaning, which can obscure certain categories of paraphrasing where the core subject matter remains constant but the descriptive content is substantially altered.

8.2 Scoring Engine Consistency

Comparison of system-generated Final Scores against the three-reviewer human panel ratings across the thirty-five original manuscripts yielded a Pearson correlation coefficient of 0.71 and a mean absolute error of 0.94 (on the ten-point scale). This moderate-to-strong correlation suggests that the weighted multi-criteria scoring formulation captures a substantial portion of the variance in human quality judgment, while the residual disagreement is consistent with the cautionary findings of Thelwall (2024) and Zhou et al. (2024) regarding the limits of automated systems in fully replicating nuanced expert judgment, particularly for criteria such as Clarity and Contribution that depend heavily on subjective interpretation.

Table. 6: Score Component Correlation with Human Panel Ratings

Component	Pearson Correlation	Mean Absolute Error
Novelty	0.68	1.12
Methodology	0.76	0.81
Clarity	0.59	1.34
Literature	0.74	0.88
Grammar	0.81	0.62
Technical Depth	0.70	0.97
Final Score (composite)	0.71	0.94

Component-level analysis reveals that structurally verifiable criteria — Grammar, Literature, and Methodology — exhibit stronger correlation with human judgment than criteria requiring more holistic or subjective assessment, such as Clarity. This pattern is expected given the heuristic, signal-based derivation of these scores described in Section 4.7, and suggests that future refinement efforts should prioritize improving the Clarity and Novelty scoring mechanisms, the two components exhibiting the weakest agreement with human raters.

8.3 Summarization Quality

Summarization performance, evaluated against author-written abstracts as reference proxies, yielded ROUGE-1, ROUGE-2, and ROUGE-L scores of 0.41, 0.19, and 0.37 respectively. These figures fall within the expected range for abstractive summarization models applied to technical scientific text, where domain-specific terminology and dense information content typically constrain achievable ROUGE scores relative to general-domain summarization benchmarks. The fallback extractive summarization mechanism was invoked in three of the fifty pilot manuscripts due to primary model unavailability during testing, in each case producing a functional, if less fluent, summary output, confirming the intended graceful-degradation.

8.4 Retrieval Efficiency

Comparison of FAISS-based nearest-neighbor retrieval against brute-force pairwise L2 distance computation

across increasing reference corpus sizes demonstrated the expected scalability advantage of indexed retrieval, summarized in Table 7.

Table. 7: Retrieval Latency: FAISS vs. Brute-Force Comparison

Corpus Size	Brute-Force Latency (ms)	FAISS Latency (ms)
100	4.2	1.8
1,000	41.6	2.3
10,000	418.7	3.1
100,000	4,193.5	4.6

The brute-force comparison exhibits approximately linear latency growth with corpus size, as expected given its exhaustive pairwise comparison strategy, whereas FAISS-based retrieval exhibits near-constant latency across a three-order-of-magnitude increase in corpus size. At the largest tested scale of 100,000 reference embeddings, FAISS retrieval was approximately 900 times faster than brute-force comparison, substantiating the efficiency rationale articulated in Section 4.5 and confirming the practical necessity of indexed retrieval for any deployment scenario involving a reference corpus of realistic scholarly scale.

8.5 Section Classification Fallback Accuracy

The SciBERT-based fallback section classifier was evaluated on a subset of eight manuscripts within the pilot corpus exhibiting non-standard or missing section headings, achieving a classification accuracy of 0.83 and a mean per-class F1-score of 0.79 across the five canonical section categories. While regex-based extraction alone failed to correctly segment these eight manuscripts, the SciBERT fallback successfully recovered usable section boundaries in all cases, confirming the practical value of the two-tier segmentation design described in Section 4.3, notwithstanding the expected reduction in classification precision relative to well-formatted manuscripts.

9. CONCLUSION

This paper described the design and implementation of a modular hybrid NLP system for automated research-paper evaluation, combining PDF ingestion and validation, section-aware summarization, semantic novelty detection through Sentence-Transformer embeddings and FAISS retrieval, deterministic multi-dimensional evaluation scoring, rule-based reviewer-feedback generation, and independent segment-level plagiarism detection within a single FastAPI-based pipeline. The principal technical contributions are the section-aware construction of the novelty-detection query, the explicit architectural and conceptual separation of whole-document novelty assessment from passage-level plagiarism detection despite their shared retrieval infrastructure, and the traceability afforded by keeping evaluation scoring and review generation deterministic and rule-based rather than delegating them to opaque model inference. Qualitative evidence from an end-to-end run of the deployed system confirms that the pipeline executes successfully on a real manuscript and produces

populated summary, review, scoring, novelty, and plagiarism outputs through its web interface; quantitative accuracy, agreement with human reviewers, and performance figures were not measured in this study and are identified explicitly as requiring controlled future evaluation rather than being estimated.

The system's practical significance lies in its potential use as a preliminary, reviewer-support screening tool that can surface structural gaps, unsupported claims, and candidate similarity concerns before a manuscript reaches a human reviewer or editor, reducing the portion of routine checking that must be performed manually, while explicitly not functioning as a substitute for expert peer judgment.

REFERENCES

1. Beltagy, I., Lo, K., & Cohan, A. (2019). SciBERT: A pretrained language model for scientific text. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP) (pp. 3615–3620). Association for Computational Linguistics.
2. Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3, 993–1022.
3. Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (pp. 4171–4186). Association for Computational Linguistics.
4. Foltyniek, T., Meuschke, N., & Gipp, B. (2019). Academic plagiarism detection: A systematic literature review. *ACM Computing Surveys*, 52(6), 1–42.
5. Johnson, J., Douze, M., & Jegou, H. (2019). Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3), 535–547.
6. Kang, D., Ammar, W., Dalvi, B., van Zuylen, M., Kohlmeier, S., Hovy, E., & Schwartz, R. (2018). A dataset of peer reviews (PeerRead): Collection, insights and NLP applications. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (pp. 1647–1661). Association for Computational Linguistics.
7. Kusner, M., Sun, Y., Kolkin, N., & Weinberger, K. (2015). From word embeddings to document distances. In Proceedings of the 32nd International Conference on Machine Learning (pp. 957–966). PMLR.
8. Liang, W., Zhang, Y., Cao, H., Wang, B., Ding, D. Y., Yang, X., Vodrahalli, K., He, S., Smith, D. S., Yin, Y., et al. (2024). Can large language models provide useful feedback on research papers? A large-scale empirical analysis. *NEJM AI*, 1(8), A10a2400196.
9. Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. In Text Summarization Branches Out (pp. 74–81). Association for Computational Linguistics.
10. Liu, R., & Shah, N. B. (2023). ReviewerGPT? An exploratory study on using large language models for paper reviewing. *arXiv preprint arXiv:2306.00622*.
11. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.
12. Malkov, Y. A., & Yashunin, D. A. (2020). Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4), 824–836.
13. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
14. Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). BLEU: A method for automatic evaluation of machine translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (pp. 311–318). Association for Computational Linguistics.
15. Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)

(pp. 1532–1543). Association for Computational Linguistics.

16. Price, S., & Flach, P. A. (2017). Computational support for academic peer review: A perspective from artificial intelligence. *Communications of the ACM*, 60(3), 70–79.

17. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2020).

Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140), 1–67.

18. Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (pp. 3982–3992). Association for Computational Linguistics.

19. Wang, Q., Zeng, Q., Huang, L., Knight, K., Ji, H., & Rajani, N. F. (2020). ReviewRobot: Explainable paper review generation based on knowledge synthesis. In *Proceedings of the 13th International Conference on Natural Language Generation* (pp. 384–397). Association for Computational Linguistics.

20. Yuan, W., Liu, P., & Neubig, G. (2022). Can we automate scientific reviewing? *Journal of Artificial Intelligence Research*, 75, 171–212.