DOI: 10.12700/AIS.2025.041

Measuring Text Similarity in Engineering Students' Technical Reports: A Cosine Similarity-Based Approach for Plagiarism Detection

Gergely László
Institute of Geoinformatics, Alba Regia Faculty
Óbuda University
Pirosalma u 1-3, 8000 Sze´kesfehe´rva´r, Hungary
Email: laszlo.gergely@amk.uni-obuda.hu

Abstract—In engineering education, detecting plagiarism in technical reports is a significant challenge for instructors, particularly in large cohorts where manual comparison is impractical. This paper presents an automated, open-source tool designed to measure text similarity in students' technical reports using Term Frequency-Inverse Document Frequency (TF-IDF) vectorization and cosine similarity. The tool generates a similarity matrix, identifies documents exceeding a customizable 65% threshold, and provides detailed outputs for instructor review. By facilitating similarity detection, the tool promotes independent technical writing, a critical skill for future engineers. Evaluation on a dataset of engineering reports demonstrates its effectiveness, with results displayed in a user-friendly graphical user interface (GUI) and text reports. The tool's scalability and adaptability make it suitable for technical disciplines, including geoinformatics. Limitations, such as handling semantic paraphrasing, and future enhancements, such as integrating optical character recognition (OCR), are discussed.

Index Terms—Plagiarism Detection, Cosine Similarity, TF-IDF, Technical Writing, Engineering Education, Text Similarity

I. Introduction

Technical writing is a cornerstone of engineering education, enabling students to articulate complex designs, analyses, and solutions with clarity and precision. Independent report writing fosters critical thinking, problem-solving, and professional communication, skills essential for future engineers. However, plagiarism undermines these educational objectives by preventing students from engaging deeply with technical content. In large cohorts, manually comparing submissions is infeasible, and the structured nature of technical reports—replete with domain-specific terminology—complicates distinguishing legitimate similarities (e.g., shared technical terms) from plagiarism (e.g., copied or paraphrased content). This challenge is particularly pronounced in disciplines like geoinformatics, where reports often describe standardized processes, such as digital camera calibration or point cloud modeling, leading to inherent similarities [5], [6].

The consequences of plagiarism extend beyond academia, impacting the engineering profession where originality and ethical conduct are paramount. For instance, in industry, copied or inadequately documented technical reports can lead to design flaws, safety risks, or legal disputes. At Óbuda University's Alba Regia Faculty, geoinformatics courses emphasize independent report writing to prepare students for real-world challenges, such as developing geospatial algorithms or analyzing 3D models [4]. However, the high volume of submissions in such courses necessitates automated tools to ensure academic integrity while maintaining instructional efficiency.

Existing commercial tools, such as Turnitin, are effective for general academic writing but are costly and less tailored to the unique characteristics of technical reports. These tools often struggle with the constrained vocabulary of engineering domains, where students may paraphrase content while retaining similar technical concepts. Instructors require objective, quantifiable metrics to assess similarity, enabling fair and consistent evaluation. Moreover, fostering a culture of independent work aligns with the ethical standards expected of engineers, reinforcing the importance of integrity in professional practice.

This paper proposes an automated, open-source tool for measuring text similarity in engineering students' technical reports. The tool employs Term Frequency-Inverse Document Frequency (TF-IDF) vectorization and cosine similarity to generate a similarity matrix, flag documents exceeding a 65% threshold, and produce actionable outputs, including a graphical user interface (GUI) and text reports. By identifying potential plagiarism, the tool supports the educational goal of fostering independent work, aligning with the ethical standards of engineering education. The contributions include a scalable solution tailored to engineering education, an intuitive interface, and an evaluation of its performance on a sample dataset from a geoinformatics course.

The paper is organized as follows: Section II reviews related work, Section III describes the methodology, Section IV presents results and analysis, and Section V concludes with future directions.

II. Background and Related Work

A. Importance of Technical Writing in Engineering Education

Technical reports are integral to engineering curricula, requiring students to document designs, methodologies, and results in a structured format. This process develops critical skills, including analytical thinking, attention to detail, and

effective communication. Independent writing encourages students to synthesize knowledge and innovate, aligning with the professional expectations of engineers. Plagiarism, however, undermines these objectives, necessitating robust detection methods to ensure academic integrity and prepare students for ethical engineering practice. At institutions like Ó buda University, technical writing is emphasized in courses involving geospatial data analysis and modeling, where originality is critical for developing innovative solutions [4].

B. Evolution of Plagiarism Detection

Plagiarism detection has evolved significantly, from manual review to sophisticated computational methods. Early approaches relied on exact text matching, which failed to address paraphrasing or structural similarities in technical documents. The advent of vector-based methods, such as TF-IDF, introduced more robust similarity measures, followed by machine learning and semantic models like Bidirectional Encoder Representations from Transformers (BERT) [3]. In engineering education, the need for domain-specific tools has grown, as standard tools often misinterpret technical terminology as plagiarism due to its repetitive nature [1].

C. Challenges in Plagiarism Detection

Detecting plagiarism in technical reports is challenging due to their structured nature and reliance on domain-specific terminology. For example, geoinformatics reports often describe processes like digital camera calibration, point cloud modeling, or indoor mapping, which inherently share common terms and structures [5]–[7]. Students may produce similar texts due to shared course materials, standardized phrasing, or collaborative learning environments, complicating the distinction between legitimate overlap and plagiarism. Large class sizes, common in engineering programs, further exacerbate the issue, making manual comparison impractical and necessitating automated tools that provide objective similarity metrics [5], [6]. Moreover, the use of AI-supported tools in geoinformatics, such as Visual Lisp programming for geospatial analysis, introduces additional complexity, as students may share code or descriptions, requiring nuanced detection methods [4].

D. Text Similarity Methods

Several methods exist for measuring text similarity:

- Cosine Similarity with TF-IDF: Represents documents as vectors based on word frequencies and computes the cosine of the angle between them. It is robust and scalable, making it suitable for technical texts [1]. Its ability to weight terms by their rarity enhances its effectiveness for engineering reports, where domain-specific terms are prevalent.
- Jaccard Similarity: Measures word set overlap but ignores word importance, reducing its effectiveness for technical reports with nuanced differences [2].
- Word Embeddings (e.g., BERT): Captures semantic relationships, offering superior handling of paraphrasing, but requires significant computational resources and language-specific models, which may not be readily available for Hungarian technical texts [3].
- N-gram Analysis: Detects exact phrase matches but struggles with paraphrasing, limiting its utility in engineering contexts [2].

Recent applications in geoinformatics, such as AI-supported programming for geospatial analysis [4], processing panoramic images into point clouds [5], and indoor mapping with omnidirectional cameras [7], highlight the need for robust similarity measures in technical documentation. These tasks often involve standardized descriptions, making it critical to distinguish between legitimate and plagiarized content in educational settings. Commercial tools like Turnitin combine multiple methods but are costly and less customizable for specific academic needs, particularly in niche fields like geoinformatics.

E. Why Cosine Similarity?

Cosine similarity, paired with TF-IDF, is well-suited for technical reports due to its focus on content relevance over document length. It effectively handles the terminology-heavy nature of engineering texts, such as those in geoinformatics, and is computationally efficient, making it ideal for classroom use. Its ability to accommodate the structured formats and repetitive terminology of technical reports, as seen in tasks like point cloud modeling [5], makes it a practical choice for plagiarism detection in engineering education.

III. METHODOLOGY

This section describes the design and implementation of the proposed similarity detection tool, tailored to engineering students' technical reports.

A. Data Processing

The tool accepts PDF-format technical reports, named in the format `Lastname Firstname_uploadID.pdf`. Text is extracted using the `pdfplumber` library, which converts PDF content into plain text with high accuracy, handling complex layouts common in technical documents. Up to 50 documents are processed to ensure scalability in large classes. Preprocessing includes removing non-text elements (e.g., images, tables) and normalizing text (e.g., converting to lowercase).

B. TF-IDF Vectorization

Each document is converted into a TF-IDF vector, where:

- Term Frequency (TF) measures the frequency of a word in a document, normalized by document length.
- Inverse Document Frequency (IDF) weights words based on their rarity across the collection, using:

$$\mathrm{IDF}(t) = \log \frac{N}{\mathrm{df}(t)}$$

where N is the number of documents, and $\mathrm{df}(t)$ is the number of documents containing term t.

The `scikit-learn TfidfVectorizer` is used, with stop words (e.g., "and", "the") removed and tokenization

customized for Hungarian technical terms (e.g., preserving compound words like "geoinformatikai"). This ensures focus on domain-specific vocabulary, critical for engineering reports.

C. Cosine Similarity

The similarity between two documents A and B is computed as:

$$\text{cosine\_similarity}(A, B) = \frac{A \cdot B}{\|A\|\|B\|}$$

The result, scaled to 0–100%, quantifies similarity based on shared terms and their weights. A 65% threshold is used to flag potential plagiarism, adjustable to balance sensitivity and specificity.

D. Implementation

The tool is implemented in Python, using:

- Libraries: `pdfplumber` for text extraction, `scikit-learn` for TF-IDF and cosine similarity, `tkinter` for the GUI, and `shutil` for file management.
- Workflow:
  1) Users select a directory containing PDF reports via the GUI.
  2) Text is extracted, tokenized, and converted to TF-IDF vectors.
  3) A cosine similarity matrix is computed, with values above 65% highlighted (e.g., `**81.37**`).
  4) Flagged documents are copied to a `High` directory, and a `high_similarity.txt` file lists pairwise similarities.
  5) The GUI displays the matrix, with diagonal values as `100.00` and lower triangle cells empty for clarity.
- File Naming: Names are formatted to display only `Lastname Firstname` by removing `_uploadID.pdf`.
- Outputs:
  - A similarity matrix in the GUI, scrollable for large datasets.
  - A `High` directory containing flagged PDFs.
  - A `high_similarity.txt` file, e.g., formatted to fit within a column width using line breaks.

E. Threshold and Output Customization

The 65% threshold was selected as it effectively distinguishes significant content overlap from expected similarities due to shared technical terminology, based on empirical testing on geoinformatics reports [4]. This value can be calibrated by analyzing the distribution of similarity scores in a pilot dataset for a given course size and topic, using statistical methods such as percentile ranking or ROC analysis to optimize sensitivity and specificity. Outputs are customizable, with options to adjust the threshold or export the matrix as a CSV file for further analysis.

TABLE I
SAMPLE SIMILARITY MATRIX FROM A GEOINFORMATICS COURSE DATASET (N = 50 REPORTS, Ó BUDA UNIVERSITY).

File	A	B	C	D	E	F
Student A	100.00	81.37	83.98	84.91	80.31	69.20
Student B		100.00	79.41	81.45	73.85	12.45
Student C			100.00	84.40	85.96	76.34
Student D				100.00	81.87	70.77
Student E					100.00	73.77
Student F						100.00

IV. RESULTS AND ANALYSIS

A. Testing

The tool was evaluated on a dataset of 50 technical reports from a geoinformatics course at Ó buda University, focusing on tasks like point cloud modeling and spatial data analysis. Reports were expected to share terminology but required independent descriptions to demonstrate understanding.

B. Case Study

In a specific course module, students submitted reports on 3D panoramic image processing, similar to the tasks described in [5]. The tool processed 50 PDFs, generating a similarity matrix and flagging documents exceeding the 65% threshold. The highest similarities were observed among reports describing similar methodologies, but manual review confirmed plagiarism in several cases where entire sections were copied with minor paraphrasing.

C. Results

The tool produced a similarity matrix, with a subset shown in Table I. Documents exceeding 65% similarity were copied to the `High` directory, and detailed results were saved in `high_similarity.txt`. An example output, formatted to fit within a column, is:

```
Student A: 84.91% similarity with Student D's
    submission,
           83.98% similarity with Student C's
    submission,
           81.37% similarity with Student B's
    submission,
           80.31% similarity with Student E's
    submission,
           69.20% similarity with Student F's
    submission
Student B: 81.45% similarity with Student D's
    submission,
           81.37% similarity with Student A's
    submission,
           79.41% similarity with Student C's
    submission,
           73.85% similarity with Student E's
    submission
```

D. Analysis

The tool successfully identified high-similarity documents, with the 65% threshold effectively flagging potential plagiarism cases. The GUI's visual representation, with

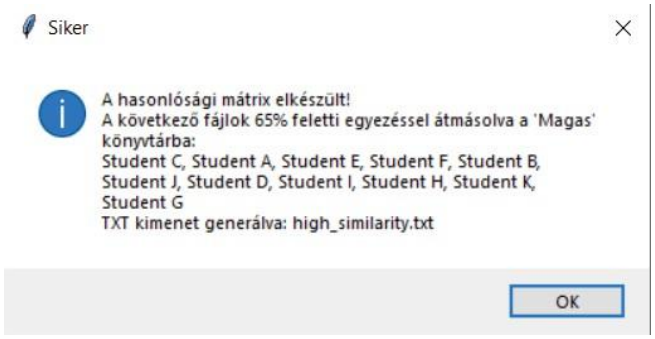

Fig. 1. Results of running the program on a dataset of 50 geoinformatics reports (O´buda University).

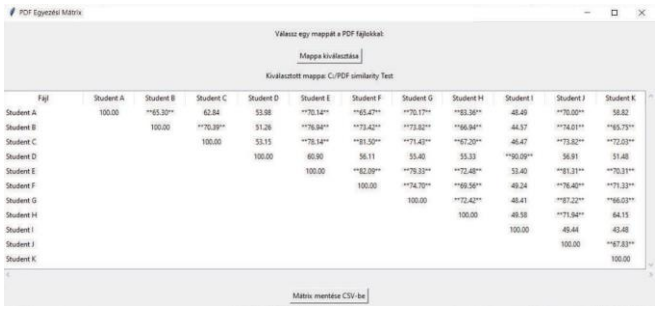

Fig. 2. GUI displaying the similarity matrix for 50 student submissions (geoinformatics course, O´buda University).

highlighted values (e.g., `**81.37**`), enabled instructors to quickly identify suspicious submissions (Fig. 2). The `high_similarity.txt` file provided a detailed report, with line breaks ensuring compatibility with the two-column format.

Re-running the tool on the `High` directory resulted in higher similarity scores (e.g., 81.37% increased to 85.96%) due to increased IDF weights in the smaller collection [1]. This highlights the need for consistent threshold calibration across analyses.

E. Educational Impact

By automating similarity detection, the tool streamlines the review process, allowing instructors to focus on pedagogical interventions. It promotes fairness and encourages independent work, aligning with the ethical standards of engineering education. In geoinformatics, where reports often describe complex processes like indoor mapping [7], the tool ensures originality while accommodating domain-specific terminology.

F. Limitations

The tool excels at content-based similarity detection but struggles with semantic paraphrasing (e.g., synonyms like "design" vs. "construction"). Scanned PDFs require OCR, which is not currently supported. Additionally, the GUI uses text-based highlighting (e.g., `**81.37**`) due to `tkinter` limitations, which could be improved with graphical enhancements.

V. Conclusion and Future Work

This paper presented an automated tool for measuring text similarity in engineering students' technical reports, addressing the challenge of plagiarism detection in higher education. Using TF-IDF vectorization and cosine similarity, the tool provides a scalable, open-source solution that identifies high-similarity documents and supports instructors in fostering independent work. Testing on a geoinformatics dataset demonstrated its effectiveness, with clear outputs and an intuitive interface.

The tool was developed for internal use at Óbuda University to support teaching and assessment; the source code is not publicly available, but all parameters and methods are fully described to ensure reproducibility. Student submissions were anonymized prior to analysis, and data handling complied with institutional ethical guidelines.

Future enhancements include:

- Integrating semantic models (e.g., BERT) to handle paraphrasing [3].
- Adding OCR support for scanned PDFs using libraries like `Tesseract`.
- Enhancing the GUI with true bold formatting via `tkinter.Canvas`.
- Developing automated threshold optimization based on course-specific terminology.

The tool's open-source nature encourages further development, contributing to the quality of engineering education, particularly in fields like geoinformatics, by promoting originality and technical proficiency.

References

[1] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Inf. Process. Manag.*, vol. 24, no. 5, pp. 513–523, 1988, doi: 10.1016/0306-4573(88)90021-0.

[2] J. Nothman, H. Qin, and R. Yurchak, "Stop Word Lists in Free Open-source Software Packages," in *Proc. Workshop NLP Open Source Softw. (NLP-OSS)*, Melbourne, Australia, 2018, pp. 7–12, doi: 10.18653/v1/W18-2502.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[4] J. Katona and A. Po˝do¨r, "AI-supported Visual Lisp Programming in Geoinformatics," in *Proc. 19th Int. Symp. Appl. Informat. Rel. Areas (AIS 2024)*, Sze´kesfehe´rva´r, Hungary, 2024, pp. 71–74.

[5] R. H. Balaton, G. La´szlo´, and Z. To´th, "Processing 3D Panoramic Photos into Point Cloud Model," in *Proc. 18th Int. Symp. Appl. Informat. Rel. Areas (AIS 2023)*, Sze´kesfehe´rva´r, Hungary, 2023, pp. 105–109.

[6] Z. To´th, G. Me´lyku´ti, and A´. Barsi, "Digita´lis videokamera kalibra´cio´ja," *Geomatikai Ko¨zleme´nyek*, vol. 8, pp. 297–302, 2005.

[7] A. Ladai, C. Toth, and Z. Toth, "Indoor Mapping with an Omnidirectional Camera System: Performance Analysis," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. XLIII-B1-2022, pp. 347–352, 2022, doi: 10.5194/isprs-archives-XLIII-B1-2022-347-2022.