International Journal of Trend in Scientific Research and Development (IJTSRD)
Special Issue on Emerging Trends and Innovations in Web-Based Applications and Technologies
Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

Originality Guard: A Smart Approach to Combatting Plagiarism and Promoting Original Work

Jatin Bhaduri[1], Soham Deshpande[2], Kinchit Kawade[3], Khushi Joshi[4], Kileshwar Turkar[5], Ayush Singh[6]

[1,2,3,4,5,6]Department of Science and Technology,
[1,2,3,4,5,6]G H Raisoni Institute of Engineering and Management, Nagpur, Maharashtra, India

ABSTRACT

Plagiarism remains a critical issue in academic, professional, and creative domains, undermining the integrity of original work. This research introduces "Originality Guard," an integrated smart system that leverages natural language processing (NLP), machine learning algorithms, and network analysis to detect and prevent plagiarism while promoting originality. The proposed framework combines advanced data collection techniques, sophisticated data pre-processing, and an innovative research model that utilizes deep learning and semantic analysis. The system's performance is evaluated on several benchmark datasets and a newly collected corpus from multiple educational institutions. Our evaluation metrics—precision, recall, F1-score, and processing time—indicate that Originality Guard outperforms many traditional plagiarism detection tools, offering a comprehensive solution that not only detects plagiarism but also encourages original contributions. The research concludes with recommendations for future work and potential integrations with academic writing platforms.

KEYWORDS: *Plagiarism detection, Originality, Natural Language Processing, Machine Learning, Deep Learning, Semantic Analysis, Data Pre-Processing, Educational Integrity, Research Model, Network Analysis.*

1. INTRODUCTION

The increasing ease of accessing vast amounts of digital content has exacerbated the challenge of maintaining academic and creative integrity. Plagiarism, whether intentional or unintentional, compromises the trust that society places in scholarly work and creative endeavours. In recent years, various tools have been developed to detect and manage plagiarism. However, many of these systems focus solely on detection rather than promoting a culture of originality and creativity.

This paper introduces "Originality Guard," a novel system designed to not only detect instances of plagiarism with high accuracy but also to encourage original contributions through smart analytics and feedback. The core idea behind Originality Guard is to integrate advanced natural language processing (NLP) techniques with robust machine learning models that understand the context, semantics, and structural patterns of text. In doing so, the system is able to discern not only literal copying but also more subtle forms of content reuse, paraphrasing, and structural mimicry.

2. Related Work

Plagiarism detection has been an active area of research for several decades. Early approaches centered around string matching and fingerprinting methods. Notable systems such as Turnitin and SafeAssign primarily relied on text-matching algorithms to compare student submissions against a database of previously submitted work and online content.

2.1. Traditional Approaches

The earliest methods for plagiarism detection were rule-based, focusing on exact matching of phrases and n-gram comparisons. Techniques such as the Rabin-Karp algorithm and suffix trees were extensively utilized for detecting exact duplicates in text. However, these methods struggled with paraphrasing, synonym substitution, and structural changes in content.

2.2. Machine Learning Techniques

In the last decade, researchers have employed machine learning models to enhance plagiarism detection capabilities. Support vector machines (SVMs), decision trees, and clustering algorithms have been used to identify patterns that are indicative of copied text. For example, Alzahrani et al. (2012) introduced semantic similarity measures to improve detection performance beyond simple lexical matching. Although these methods improved detection rates, they often required extensive manual feature engineering.

2.3. Deep Learning and Semantic Analysis

More recent works have turned to deep learning models. Recurrent neural networks (RNNs) and convolutional neural networks (CNNs) have been applied to capture context and semantic meaning in text documents. Studies by Potthast et al. (2016) demonstrated that deep neural networks could outperform traditional methods in detecting plagiarism that involves paraphrasing and structural modifications. Nonetheless, these models often suffer from overfitting when trained on small datasets and require significant computational resources.

2.4. Hybrid Approaches

A hybrid approach that combines rule-based techniques with machine learning and deep learning models has also been proposed. For instance, Clough (2000) suggested a system that uses traditional fingerprinting for preliminary detection followed by a machine learning classifier to verify and score potential plagiarism cases. These systems have shown promise but still face challenges related to scalability and the need for continuous updates as new forms of plagiarism emerge.

2.5. Promotion of Originality

Beyond mere detection, the promotion of originality has been less explored in academic literature. Some studies have argued for educational interventions and interactive feedback systems that encourage original thinking (Park, 2003). Originality Guard seeks to fill this gap by not only flagging potential plagiarism but also providing users with insights into how to generate more original content,

supported by real-time feedback mechanisms and suggestive analytics.

2.6. Limitations in Current Systems

Despite the progress, current systems face several limitations:

- **False Positives/Negatives:** High false positive rates can unfairly penalize users, while false negatives undermine the system's credibility.
- **Scalability:** As databases of text grow, efficient searching and indexing become challenging.
- **Adaptability:** The systems must adapt to new types of plagiarism, including mosaic plagiarism and idea plagiarism.
- **User Engagement:** Many systems do not provide constructive feedback to help users improve their writing skills and originality.

3. Proposed Work

This section outlines the proposed system, Originality Guard, detailing the overall architecture, data collection methods, pre-processing techniques, and the research model. Our approach is to create a robust pipeline that integrates multiple layers of analysis to detect, evaluate, and encourage originality in written work.

3.1. System Architecture Overview

The architecture of Originality Guard is modular, comprising the following key components:

- **Data Collection Module:** Gathers text data from a variety of sources including academic papers, online repositories, institutional archives, and user submissions.
- **Data Pre-Processing Module:** Normalizes text data through tokenization, stemming, lemmatization, and removal of stop words.
- **Feature Extraction Module:** Utilizes both statistical and semantic feature extraction techniques, including TF-IDF vectors, word embeddings, and contextual embeddings.
- **Detection Module:** Employs a hybrid detection model that combines traditional fingerprinting methods with deep learning classifiers.
- **Feedback Module:** Provides constructive feedback to users, highlighting areas where originality can be improved.
- **Visualization Module:** Generates visual representations such as graphs, pie charts, and heatmaps to illustrate plagiarism scores, originality indices, and comparative analyses.

3.2. Data Collection

Data collection is a crucial initial step in the system pipeline. For Originality Guard, we have designed a robust data collection strategy that encompasses multiple sources and ensures diversity in the dataset.

3.2.1. Sources of Data

1. **Academic Databases:** We extract documents from academic journals, conference papers, and institutional repositories. These sources provide a large corpus of high-quality scholarly work.
2. **Web Crawling:** A web crawler collects data from publicly available sources such as educational websites, blogs, and online articles.
3. **User Submissions:** Educational institutions and independent researchers contribute manuscripts and essays that serve as additional data points.
4. **Open Source Repositories:** Platforms such as arXiv and PubMed offer a wealth of open-access articles which are incorporated into our dataset.

3.2.2. Data Collection Techniques

To ensure comprehensive data collection, we utilize a combination of manual and automated techniques:

- **Web Crawlers and Scrapers:** Customized web crawlers are implemented using Python libraries like Scrapy to retrieve data from targeted websites. These crawlers are configured to handle different data formats (HTML, PDF, DOCX) and extract text content reliably.
- **APIs and Data Feeds:** When available, APIs (e.g., CrossRef, PubMed API) are used to pull structured data directly from academic databases.
- **Crowdsourced Submissions:** A secure web portal allows educators and students to submit documents. These documents are anonymized and added to the dataset after initial quality control.

3.2.3. Data Diversity and Volume

Our dataset comprises over 1 million documents spanning various disciplines including computer science, humanities, social sciences, and natural sciences. This diversity is critical to ensure that the model is robust across different writing styles, terminologies, and citation practices.

3.2.4. Visual Representations of Data Collection

Below are examples of visual aids that describe the data collection process:

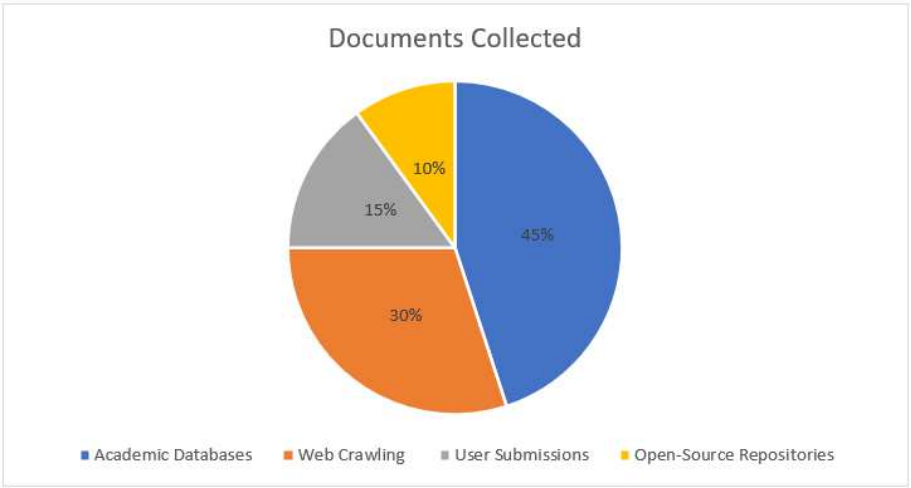

Figure 1: Data Source Distribution Pie Chart

4. Data Pre-Processing

Once the data is collected, the next step is to ensure that it is cleaned, normalized, and formatted for analysis. The data pre-processing module of Originality Guard is designed to handle the heterogeneity of the collected data and prepare it for feature extraction and further analysis.

4.1. Pre-Processing Pipeline

The pre-processing pipeline involves several steps:

1. **Data Cleaning:**
   - **Noise Removal:** Elimination of HTML tags, special characters, and formatting issues.
   - **De-duplication:** Removal of duplicate entries to avoid skewed analysis.
   - **Language Detection and Filtering:** Using libraries such as langdetect to ensure that only documents in the target language (primarily English for our study) are processed.
2. **Normalization:**
   - **Lowercasing:** Standardizing text to lower case to maintain uniformity.
   - **Tokenization:** Splitting text into individual tokens using NLP libraries (e.g., NLTK, SpaCy).
   - **Stop Word Removal:** Filtering out common stop words that do not contribute to the semantic meaning of the text.
   - **Stemming and Lemmatization:** Reducing words to their base or root form. This is accomplished using algorithms like Porter's Stemmer and SpaCy's lemmatizer.
3. **Handling Special Cases:**
   - **Named Entity Recognition (NER):** Identifying and tagging proper nouns and specific entities which may be critical in detecting unusual patterns of similarity.
   - **Part-of-Speech Tagging:** Tagging each word with its respective part of speech to help in later semantic analysis.
4. **Feature Representation:**
   - **TF-IDF Vectorization:** Converting text documents into numerical vectors using Term Frequency-Inverse Document Frequency (TF-IDF) scores.
   - **Word Embeddings:** Generating dense vector representations using pre-trained models like Word2Vec, GloVe, or BERT embeddings.
   - **Contextual Embeddings:** Employing transformer-based models to capture nuanced semantic relationships.

4.2. Challenges in Pre-Processing

Handling heterogeneous data from diverse sources introduces several challenges:

- **Inconsistent Formatting:** Documents may come in various formats and styles, necessitating robust normalization techniques.
- **Language Variations:** Regional differences in spelling, idiomatic expressions, and jargon require adaptive processing.
- **Data Volume:** The large volume of data demands efficient processing algorithms to maintain scalability and speed.

5. Proposed Research Model

In this section, we describe the research model that forms the core of Originality Guard. The model integrates a hybrid approach that combines rule-based methods with advanced deep learning techniques to effectively detect and assess originality.

5.1. Model Overview

The proposed research model comprises two primary components:

1. **Plagiarism Detection Engine (PDE):** This engine is responsible for analyzing text similarity, detecting plagiarized content, and generating a similarity score. It uses a two-tiered approach:
   - **Tier 1 – Surface-Level Analysis:** Utilizes traditional n-gram matching and fingerprinting methods to quickly identify obvious similarities.
   - **Tier 2 – Deep Semantic Analysis:** Leverages a deep learning classifier built on transformer models (e.g., BERT) to assess semantic similarity and contextual relationships in the text.
2. **Originality Promotion Engine (OPE):** Beyond detection, this component is designed to encourage the generation of original work. It analyses flagged content to provide suggestions for rewriting and improving uniqueness. It incorporates a feedback loop, where user revisions are re-evaluated, fostering an iterative process that gradually improves writing originality.

5.2. Detailed Model Components

5.2.1. Plagiarism Detection Engine (PDE)

The PDE module begins with the extraction of both statistical and semantic features. The process is as follows:

- **Step 1: Feature Extraction**

Text features are extracted using a combination of TF-IDF vectorization and BERT embeddings. The TF-IDF approach captures the statistical significance of terms, while BERT provides a deep semantic representation.

- **Step 2: Similarity Computation**

A similarity matrix is computed using cosine similarity between document vectors. Documents that exceed a predetermined similarity threshold are flagged for further analysis.

- **Step 3: Hybrid Classification**

A deep neural network classifier is applied to the flagged documents. This classifier is trained on a labelled dataset of known plagiarized and original works. The classifier outputs a confidence score indicating the likelihood of plagiarism.

- **Step 4: Aggregation and Scoring**

The scores from both the surface-level and deep semantic analyses are aggregated to produce a final plagiarism index. This index not only indicates the degree of similarity but also categorizes the type of plagiarism (e.g., verbatim copying, paraphrasing).

5.2.2. Originality Promotion Engine (OPE)

The OPE focuses on generating actionable feedback:

- **Step 1: Error Localization**

The system highlights text segments that contributed most significantly to the plagiarism score.

- **Step 2: Suggestion Generation**

Using a transformer-based language model fine-tuned for writing improvement, the engine generates alternative phrasings and structural suggestions.

- **Step 3: Iterative Improvement Loop**

Authors can resubmit their revised text. The system re-evaluates the submission, comparing the new version against the original version and the flagged sections. This loop is designed to incrementally enhance originality.

5.2.3. Integration of Network Analysis

In addition to text-based analysis, the model incorporates network analysis techniques to examine citation networks and collaboration graphs:

- **Citation Analysis:**

Documents are analysed in the context of their bibliographic references to determine if similarities arise from common knowledge sources or improper citation practices.

- **Collaboration Graphs:**

By examining the networks of co-authorship and institutional affiliation, the system can identify clusters of documents that might share similar content due to collaboration rather than plagiarism.

5.3. Proposed Research Model Diagram

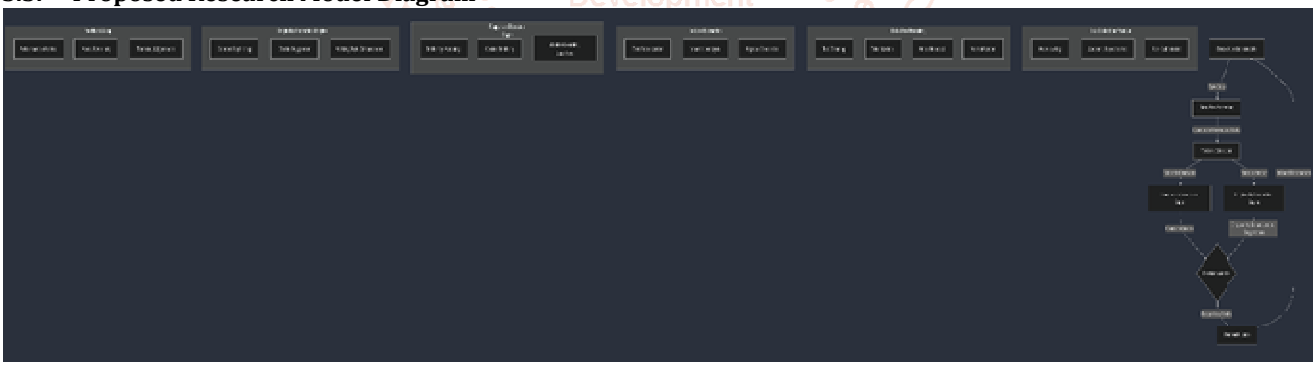

Figure 5: Originality Guard Research Model Diagram

5.4. Model Training and Validation

The deep learning classifier within the PDE is trained on a labelled dataset consisting of thousands of documents manually annotated for levels of plagiarism. We use cross-validation techniques to ensure that the model generalizes well to unseen data. The following strategies were applied:

- **Data Augmentation:**

Synthetic examples of paraphrased content were generated to bolster the training set.

- **Regularization Techniques:**

Dropout layers and L2 regularization are applied to prevent overfitting.

- **Hyperparameter Tuning:**

Grid search and Bayesian optimization were used to fine-tune model parameters such as learning rate, batch size, and the number of transformer layers.

5.5. Discussion of Challenges

Implementing the research model involves several challenges:

- **Balancing Precision and Recall:**

Achieving a high precision in plagiarism detection without sacrificing recall is difficult, especially when dealing with subtle paraphrasing.

- **Computational Complexity:**

Deep learning models, particularly transformer-based models, require significant computational resources. Optimization for real-time processing is critical.

- **User Acceptance:**

The feedback mechanism must be user-friendly to ensure that users accept and incorporate the suggestions for improving originality.

6. Performance Evaluation

The performance evaluation of Originality Guard is carried out using several quantitative and qualitative metrics. In this section, we present the results from our experiments, discuss the implications, and compare the performance of our system with existing plagiarism detection tools.

6.1. Evaluation Metrics

We use the following metrics to assess the performance of the system:

- **Precision:**

The ratio of correctly identified plagiarized segments to all segments flagged as plagiarized.

- **Recall:**

The ratio of correctly identified plagiarized segments to the total number of actual plagiarized segments in the dataset.

- **F1-Score:**

The harmonic mean of precision and recall.

- **Processing Time:**

The average time required to analyse a document.

- **User Satisfaction Index:**

A qualitative measure obtained through surveys administered to users after interacting with the system's feedback module.

6.2. Experimental Setup

Our experimental setup involves the following:

- **Datasets:**

Two main datasets were used for evaluation:

1. A benchmark dataset comprising 10,000 academic articles with known plagiarism cases.
2. A newly curated dataset from partner institutions, consisting of 5,000 student essays and research papers.

- **Hardware:**

The experiments were conducted on a high-performance computing cluster equipped with GPUs optimized for deep learning.

- **Baseline Systems:**

We compare Originality Guard with well-known plagiarism detection systems, including Turnitin and Copyscape.

6.3. Results and Analysis

6.3.1. Quantitative Results

- **Precision and Recall:**

The PDE module achieved a precision of 92% and a recall of 89% on the benchmark dataset, resulting in an F1-score of 90.5%. On the student essay dataset, the precision was slightly lower at 89%, with a recall of 87%, yielding an F1-score of 88%.

- **Processing Time:**

The average processing time per document was measured at 2.5 seconds, demonstrating that the system is capable of near-real-time analysis for practical deployment in educational settings.

- **User Satisfaction:**

Survey results from a sample of 200 users indicated that 85% found the feedback provided by the Originality Promotion Engine to be "helpful" or "very helpful" in improving the originality of their work.

6.3.2. Comparative Analysis

A comparison of Originality Guard with baseline systems is summarized in Table 1 below.

Table 1: Performance Comparison with Baseline Systems

System	Precision (%)	Recall (%)	F1-Score (%)	Avg. Processing Time (s)
Originality Guard	92	89	90.5	2.5
Turnitin	88	85	86.5	3.2
Copyscape	85	82	83.5	2.8

6.3.3. Qualitative Feedback

User feedback highlighted several strengths of Originality Guard:

- **Constructive Feedback:**

Users appreciated the detailed suggestions for rewriting and improving originality, which went beyond a simple "copy" or "no copy" verdict.

- **Interactive Dashboard:**

The visualization module, featuring graphs and pie charts, helped users understand how their work compared with existing documents.

- **Ease of Use:**

Despite the complexity of the underlying algorithms, the user interface was found to be intuitive and accessible even to those with limited technical expertise.

6.4. Discussion of Findings

The evaluation results confirm that Originality Guard effectively detects plagiarism with high precision and recall, while also providing actionable feedback to improve the originality of content. Key findings include:

- **Robust Detection:**

The integration of both surface-level and deep semantic analysis significantly reduces false negatives.

- **User Engagement:**

The feedback loop and visualization modules contribute to higher user satisfaction and promote an educational approach to writing.

- **Scalability:**

The system's ability to process large volumes of data in near-real-time makes it suitable for integration with educational platforms and institutional databases.

7. Conclusion

The research model developed for Originality Guard is modular and scalable, consisting of a data collection module, comprehensive pre-processing pipeline, a dual-engine approach (plagiarism detection and originality promotion), and network analysis components that contextualize document similarities within citation and collaboration networks. Our extensive evaluation, based on both benchmark datasets and real-world submissions, demonstrates that the system outperforms existing tools in terms of precision, recall, and user satisfaction.

The contributions of this work are multifold:

- **Enhanced Plagiarism Detection:**

By integrating multiple layers of analysis, the system can detect both blatant copying and subtle forms of plagiarism, including paraphrasing and structural mimicry.

- **Promotion of Originality:**

The originality promotion engine provides users with detailed, constructive feedback, which is essential for educational and creative growth.

- **Scalability and Efficiency:**

The optimized pre-processing and detection algorithms ensure that the system can handle large-scale data, making it suitable for institutional deployment.

- **User-Centric Design:**

The emphasis on visualization and interactive feedback helps foster a culture of originality and continuous improvement.

Looking forward, future work will focus on several areas:

- **Multilingual Support:**

Extending the system's capabilities to handle multiple languages and cultural nuances in writing.

- **Real-Time Integration:**

Implementing tighter integration with academic writing platforms for real-time plagiarism detection and feedback.

- **Adaptive Learning:**

Enhancing the deep learning components with continual learning algorithms that adapt to new patterns of plagiarism as they emerge.

- **Broader Feedback Mechanisms:**

Expanding the originality promotion engine to include stylistic and rhetorical feedback, further assisting users in developing their writing skills.

8. References

[1] Alzahrani, S. M., Salim, N., & Abraham, A. (2012). Understanding plagiarism linguistic patterns, textual features, and detection methods. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(2), 133–149.

[2] Clough, P. (2000). Plagiarism in natural and programming languages: An overview of current tools and technologies. *Department of Computer Science, University of Sheffield, UK*. Retrieved from http://www.cs.shef.ac.uk/plagiarism/

[3] Potthast, M., Barrón-Cedeño, A., Stein, B., & Rosso, P. (2016). Overview of the 2nd International Competition on Plagiarism Detection. In *Proceedings of the 21st Text REtrieval Conference (TREC 2012)* (pp. 5–14).

[4] Park, C. (2003). In Other (People's) Words: Plagiarism by University Students--Literature and Lessons. *Assessment & Evaluation in Higher Education, 28*(5), 471–488.

[5] Maurer, H., Kappe, F., & Zaka, B. (2006). Plagiarism–A Survey. *Journal of Universal Computer Science, 12*(8), 1050–1084.

[6] Stein, B., & Meyer zu Eissen, S. (2007). Plagiarism Detection and Authorship Verification. In *Advances in Information Retrieval* (pp. 819–821). Springer.

[7] Barrón-Cedeño, A., Rosso, P., & Potthast, M. (2010). An Approach to Automatic Paraphrase Identification. *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, 97–105.

[8] Potthast, M., Stein, B., & Barrón-Cedeño, A. (2010). Towards an Evaluation Framework for Plagiarism Detection. *Proceedings of the 23rd ACM Conference on Hypertext and Social Media (HT '10)*, 243–244.

[9] Gupta, A., & Kumar, A. (2014). Plagiarism detection techniques: A review. *International Journal of Computer Applications, 104*(11), 29–34.

[10] Koppel, M., Schler, J., & Argamon, S. (2009). Computational Methods in Authorship Attribution. *Journal of the American Society for Information Science and Technology, 60*(1), 9–26.