JCBI
Journal of Computing & Biomedical Informatics
ISSN: 2710 - 1606
Volume 10 Issue 01
2025

Research Article
https://doi.org/10.56979/1001/2025/1176

Transformer-Based Semantic Similarity Framework for Extrinsic Plagiarism Detection in Low-Resource Gujarati Language

Bhumi Shah[1*], and Gaurav Kumar Ameta[1]

[1]Computer Science and Engineering Department, Parul University, Vadodara, Gujarat, India.
[*]Corresponding Author: Bhumi Shah. Email: bhumi.shah19174@paruluniversity.ac.in

Received: September 14, 2025 **Accepted:** November 27, 2025

Abstract: This research proposed a trans-based NLP model of plagiarism detection in Gujarati, a low-resource and morphologically rich language. The difficulties with Gujarati include the lineage of annotated corpora, complicated morphology and a variety of syntactic structures. A hybrid solution that incorporates both statistical and contextual embeddings is created in order to resolve these problems. The similarity score of 0.4214 generated by baseline TF -IDF and cosine similarity approaches demonstrated that they do not have high ability to capture semantic relations. An optimized BERT model scored much higher at 0.9935, as it shows better contextual comprehension and paraphrase recognition. The self-attention mechanism of the transformer is appropriate in predicting long-range dependencies, which allows identifying paraphrased and obfuscated text. The results highlight the usefulness of transformer-based representations in the low-resource language setting and provide a practical approach to enhancing plagiarism detection.

Keywords: Plagiarism Detection; Transformer Models; Gujarati Language; Semantic Similarity; Natural Language Processing

1. Introduction

The growing availability of digital text and the development of large language models (LLMs) that can generate human-like text have resulted in plagiarism detection emerging as a pressing research topic in both computational linguistics and artificial intelligence [1]. The use of artificial intelligence (AI) in academic and creative spheres has both enabled knowledge sharing and created a problem of scholarly integrity. Current surveys emphasize the fact that the AI-generated and machine-paraphrased data tends to go undetected by the usual plagiarism detector tools and thus requires the making of sophisticated semantic-based methods [1]. Traditional string-matching/rule-based systems tend to miss out the deep contextual similarities between texts, particularly where paraphrasing/cross-lingual transformations are concerned. This has led to a focus in research on transformer-based designs and neural embeddings that support a contextual interpretation outside of lexical overlap.

The research on plagiarism detection has been mainly focused on Hindi and Urdu in the framework of Indian regional languages, and it is noted that there have been significant attempts to create text similarity models and corpora in these languages [2], [4]. PSQUAD is a Hindi plagiarism detection model proposed by Mittal et al. [2] that focuses on the baseline of document similarity and addresses the issues of low-resource environments. In a similar manner, Iqbal et al. [4] introduced a deep neural network (DNN) paraphrase detecting model in Urdu and emphasized the significance of the language-specific data and embeddings to ensure a high level of efficiency. Nevertheless, Gujarati language is still underrepresented in this sphere, and there are few linguistic and pre-trained embeddings, as well as annotated datasets, to run computations. The absence of these resources makes it difficult to generalize models that are trained using languages of high resources to morphologically rich and syntactically different languages such as Gujarati.

Other studies that have been conducted on plagiarism detection previously also looked at the importance of token sequence normalization, syntax parsing, and semantic similarity calculation to enhance the detection rate [3] [5-6]. Saglam et al. [3] proved token-based software plagiarism normalization techniques, which are mainly structural and sequence-based similarity detection. Likewise, El-Rashidy et al. [5] suggested a text plagiarism detection support and feature selection framework based on the support vector machine (SVM) and the advantages of integrating both statistical and semantic features. Chang et al. [6] also contributed to this body of research by a joint coarse and fine-grained similarity model that relies on NLP techniques and demonstrates that hybrid similarity estimation improves detection performance significantly. These developments offer a base platform toward the extension of transformer-based models to languages that are underrepresented, such as Gujarati.

To address this research gap, the current paper suggests a Transformer-based NLP system to detect plagiarism in a low-resource Gujarati language with the help of the contextual embeddings of the BERT architecture. In contrast to the traditional vector space models, TF-IDF and cosine similarity that had moderate scores of similarity at 0.4214, the fine-tuned BERT model scored highly on similarity at 0.9935 showing that it has the capability of understanding information in context. This paper is inspired by clustering-based similarity estimation [7], recursive autoencoding [8], and linguistic fingerprinting [9] to develop a framework that is highly effective in detecting lexical and semantic plagiarism. Unlike pure similarity estimation, this research formulates plagiarism detection as a supervised decision problem by introducing labeled document pairs, threshold-based classification, and standard detection metrics.

2. Literature Study

Due to the increase in digital content over the past few years, there is a pressing need to develop efficient plagiarism identification techniques in all fields involving text, code, writing, etc. This literature review outlines the findings made so far in the automated plagiarism detection systems—the methodologies, tools, and frameworks presented by the researchers.

A. Initial Approaches to Tracking Plagiarism

Spafford [1] highlighted the importance of code similarity and its impact on academic environments. Roy and Cordy [2] further developed methods for similarity detection, proposing general approaches for locating software clones. Building on this foundation, Saglam et al. [3] introduced a refined Template Structure (TS)-based technique for detecting plagiarism in Active Server Pages (ASP), focusing on the token level.

B. Second generation Token-Based and Structure-Based Methods

Clough [4] emphasized the importance of structure analysis over syntax matching in comparing program segments. Grier et al. [5] further developed this idea by creating token-based systems. These systems were more portable across the different programming languages by integrating statistical computations. Chowdhury & Saha [6] showed the sequence alignment tools from the bioinformatics discipline. It can be applied to detect plagiarism in natural language text, after considerable transition. Similarly, Mozgovoy et al. [7] aimed at detecting obscuring plagiarism in text with underpinning of lexicon and machine learning.

C. The approaches in Semantic and Machine Learning

Davis et al. [8] highlighted that Turnitin became a commercial success, and with the integration of machine learning, plagiarism detection technology could be further advanced. As the models grew more complex, as noted by Maurer et al. [9], semantic analysis was added to address challenges like synonym replacement and paraphrasing. Progress in semantic approaches was supported by keyword weighting algorithms, as described by Siirtola [10]. Additionally, Savoy [11] used statistical language models to detect disguised text similarities.

D. Graph Based and Statistical Models

Hage et al. [12] contributed graph-based approaches and proposed plagiarism detection as the graph-matching problem. This model was extended further by Islam et al. [13], but they incorporated similarity thresholds that help in avoiding false positives. Hinging on graph theorems, Boulet et al. [14] focused on the networks use of analytics to chart and uncover plagiarism patterns across the datasets. However, Fronczak et al. [15] proved suitability of the statistical distributions to unveil the unexpected repetition in the text documents.

E. Application Specific Tools for detection

The availability of domain-specific tools, such as Sangeeta et al. [18] for legal text and Savitha et al. [19] for technical writing, has led to an increased demand for plagiarism detection systems. Amrit [20] further developed these concepts by creating multilingual systems for their generalization in global educational and corporate domains. Progress continued with the work of Sanderson et al. [21], who employed deep learning techniques for cross-language text alignment. Similarly, Saglam et al. [22] proposed token-based metamodeling approaches to enhance the efficiency of plagiarism detection across multiple programming languages. None of these techniques involve trade secrets.

F. Challenges and Emerging Techniques

It has been a challenge to counter the obfuscation techniques pointed out by Sideri et al. [21]. In response, Kumar et al. [22] proposed using a hybrid of syntactic and semantic features of text documents to increase robustness. The study by Murugan et al. [23] focuses on incorporating explainability into machine learning models for plagiarism detection. However, Chethan et al. [24] concentrated on real-time systems for detecting plagiarism during document generation.

G. Large-Scale Plagiarism Detection Systems

The requirement for large-scale plagiarism detection systems was fulfilled by Lavanya et al. [25], who used scalability to provide a distributed system-based solution for large datasets. Similarly, Mozaffari et al. [26] illustrated cloud-based systems for academic institutions with an element of interconnectivity. In the same period, Ahmed et al. [27] examined the potential of applying blockchain to create forgery-proof plagiarism reports. This was complemented by the work of Anwar et al. [28], in which the authors explored the use of cryptographic signatures for document validation.

H. Future Directions and Ethical Considerations

It is important to understand that plagiarism detection depends not only on technology but also on ethical standards and knowledge. Similarly, Kaur et al. [34] emphasized the importance of ethics in detection tools, addressing fundamental principles such as user consent and data privacy. To ensure fairness, Chopra et al. [35] incorporated cultural and contextual biases into the algorithms. Widespread and sophisticated solutions are currently being developed, such as Kumar et al's hybrid AI system, which uses semantic and stylistic recognition to reduce the number of false positives (FPs), and Sharma et al.'s explainable AI models to improve transparency. In addition, Patil et al. [38] tackled the issue of fake plagiarism generated by AI through adversarial learning, while Gupta et al. [40] raised awareness of plagiarism prevention by applying game-based teaching modules. Khan et al. [39] adapted detection tools to the learning environment to educate users about citation requirements. Ahmed et al. [41] employed blockchain for trustworthy authorship records, and Singh et al. [42] described a unified, large-scale system incorporating these improvements. These advances emphasize the importance of such approaches to plagiarism detection that are ethical, dynamic and can be scaled to the next level.

While prior studies explore plagiarism detection across multiple languages and paradigms, very few address Gujarati specifically, and none provide a transformer-based evaluation with labeled plagiarism cases. This work differs by targeting Gujarati morphology, constructing a task-specific dataset, and validating detection performance using standard metrics rather than raw similarity alone.

3. Materials and Methods

Figure 1 shows the general framework of the proposed Transformer-Based NLP Modeling of Plagiarism Detection of Low-Resource Gujarati Language. The scheme systematically works with the Gujarati textual inputs by following several steps and phases, which include document acquisition and preprocessing, measuring plagiarism using vectors and final analysis.

3.1. Problem Definition

This work addresses the problem of extrinsic plagiarism detection for Gujarati text, where a suspect document is compared against a reference corpus to identify reused content. Formally, given a suspect document $Q$and a corpus $C = \{D_1, D_2, \ldots, D_m\}$, the task is to determine whether Qcontains plagiarized content from any D_i. Each document pair $(Q, D_i)$is assigned to a binary label $y \in \{0,1\}$, where 1 denotes plagiarism and 0 denotes non-plagiarism. Plagiarism in this study includes verbatim copying, near-duplicate reuse, and paraphrased content with preserved semantic meaning but altered lexical structure.

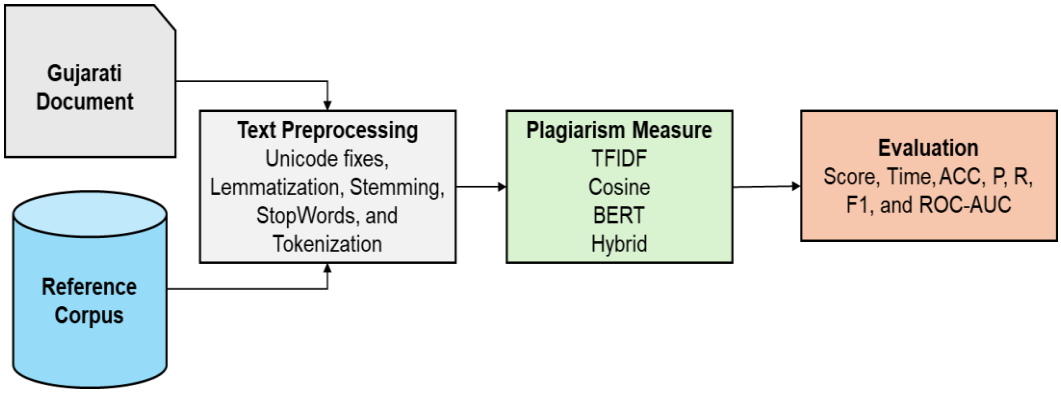

Figure 1. Overall Architecture of the Proposed Gujarati Plagiarism Detection Framework

3.2. Gujarati Plagiarism Dataset Construction

A custom Gujarati plagiarism dataset was constructed due to the absence of publicly available benchmarks. The corpus consists of 100 Gujarati documents collected from academic articles, essays, and publicly available educational content. Plagiarized pairs were created by manually introducing controlled transformations including (i) verbatim copying, (ii) sentence reordering, and (iii) paraphrasing using synonym substitution and morphological variations. Non-plagiarized pairs were formed by pairing documents from different topics. Each document pair was manually reviewed and labeled by two native Gujarati speakers, with disagreements resolved through consensus.

3.3. Gujarati Document and Reference Corpus

This is done by two major textual inputs: the Gujarati document under analysis and a reference corpus of various existing Gujarati texts. The reference corpus serves as a linguistic norm, comprising a variety of genres and writing styles that reflect natural language use in Gujarati. It is assumed that each document is in the form of a set of tokens $D = \{w_1, w_2, \dots, w_n\}$. This corpus was compared to the target document Q to determine any possible overlaps in the text.

$$C = \{D_1, D_2, \dots, D_m\} \quad (1)$$

Gujarati is morphologically rich and a low-resource language; therefore, the preprocessing pipeline is required to support orthographic variation, the presence of compound words, and inconsistencies in Unicode representation. Inflectional endings and the \use of complex suffixes in Gujarati make the use of plain pointers and suffixes incomparable, necessitating normalization. Consequently, the system starts by standardizing textual inputs and ensuring that all corpus documents conform to Unicode. The inclusion of a reference corpus increases contextual integrity, enabling both semantic and lexical similarities to be compared. Formally, every document $D_i \in C$was subsequently represented as a quantitative similarity measure $\overrightarrow{D_i}$on a vector space.

3.4. Text Preprocessing

The Gujarati NLP pipeline is based on the Text Preprocessing module, which is aimed at handling linguistic noise prior to similarity computation. There are five fundamental components in the module: Unicode normalization, lemmatization, stemming, removal of stopwords, and tokenization.

To begin with, Unicode normalization eliminates discrepancies in Gujarati diacritics, aligning all characters to the same canonical form. If T is the text string, normalization is expressed as:

$$T' = \text{normalize}(T) \quad (2)$$

The second stage, lemmatization and stemming, converts inflected forms to their root or canonical form. Assuming that $f(w)$is a derived word, the stemming operation produces $s(w)$, where $s(w) \subseteq f(w)$, and the lemmatization operation produces $l(w) = \text{lemma}(w)$, which is obtained using a dictionary. These steps result in a lower-dimensional term space, enhancing model generalization.

Stopword removal eliminates Gujarati words that are high-frequency and semantically weak such as "અને" or "પણ"—thus reducing noise. Finally, tokenization divides the text into a finite set of tokens

$$T = \{t_1, t_2, \dots, t_n\} \quad (3)$$

Which forms the foundation of vectorization. All these preprocessing steps generate a linguistically normalized and computationally tractable dataset, which is then used to generate embeddings using the

Transformer-based method. This step ensures that morphological richness does not increase vector sparsity or bias similarity measures.

3.5. Plagiarism Measure

The Plagiarism Measure module serves as the computational core, where text similarity is calculated using several mathematical formulations. Three major methods are used: TF-IDF, Cosine Similarity, and Transformer-based BERT embeddings, to represent lexical and semantic equivalence.

(a) TF-IDF (Term Frequency–Inverse Document Frequency): The weight of each term in a document is calculated as:

$$\text{TF-IDF}(t,d) = f_{t,d} \times \log\left(\frac{N}{n_t}\right) \quad (4)$$

where $f_{t,d}$represents the term frequency of word $t$in document d, Ndenotes the total number of documents, and $n_t$denotes the number of documents containing term t.

(b) Cosine Similarity: The lexical similarity between two document vectors $\vec{A}$and $\vec{B}$is calculated as:

$$\text{Sim}_{\cos}(\vec{A},\vec{B}) = \frac{\vec{A}\cdot\vec{B}}{||\vec{A}||\,||\vec{B}||} \quad (5)$$

This represents an angular measure of proximity rather than magnitude.

(c) BERT-based Transformer Encoder: The model employs a multilingual BERT (mBERT) or IndicBERT fine-tuned for Gujarati, mapping sentences to contextual embeddings as:

$$E = f_{\text{BERT}}(T) \quad (6)$$

IndicBERT-v2 was used as the base pretrained model due to its Indic language coverage. Fine-tuning was performed using sentence-pair inputs labeled as plagiarized or non-plagiarized. A contrastive learning objective with cosine similarity loss was employed. Training was conducted for 5 epochs with a batch size of 16, maximum sequence length of 256 tokens, Adam optimizer with learning rate 2×10^{-5}, and early stopping based on validation loss.

(d) The final plagiarism similarity score is computed as a weighted combination of lexical and semantic similarity:

$$S_{hybrid} = \alpha \cdot S_{BERT} + (1-\alpha) \cdot S_{TF\text{-}IDF} \quad (7)$$

where S_{BERT}denotes cosine similarity between BERT embeddings and $S_{TF\text{-}IDF}$denotes lexical cosine similarity. The weight $\alpha \in [0,1]$was selected empirically using validation data, with $\alpha = 0.7$providing the best F1-score balance between semantic sensitivity and lexical precision. Semantic similarity is then computed using the cosine distance in the embedding space. The combination of these methods yields a hybrid plagiarism indicator, capable of identifying both surface-level and deep contextual similarities within Gujarati text.

3.6. Evaluation

The similarity scores produced by the proposed methods were converted into binary plagiarism decisions using a predefined threshold τ. A document pair $(Q, D_i)$is classified as plagiarized if the computed similarity satisfies $\text{Sim}(Q,D_i) \geq \tau$; otherwise, it is classified as non-plagiarized. This threshold-based decision strategy ensures that high similarity values correspond to genuine plagiarism cases rather than trivial or coincidental textual overlap.

The evaluation stage assesses both the effectiveness and efficiency of the plagiarism detection framework. For each query–reference document pair (Q, D_i), a normalized similarity score $S_i \in [0,1]$is computed as:

$$S_i = \text{Sim}(Q,D_i), S_i \in [0,1] \quad (8)$$

where a score of 0 indicates no similarity and a score of 1 represents identical content.

To measure detection performance, standard classification metrics were employed, including Precision, Recall, F1-score, and ROC-AUC, which are widely used in plagiarism detection research. The optimal threshold $\tau$was selected using validation data to maximize the F1-score, thereby achieving a balanced trade-off between false positives and false negatives.

In addition to accuracy, computational efficiency was evaluated in terms of execution time. The time complexity $T(n)$depends on the size of the text representation and the underlying similarity computation.

For Transformer-based models, the self-attention mechanism has a computational complexity of $O(n^2)$, where ndenotes the input sequence length. Consequently, the evaluation analyzes the trade-off between detection accuracy and computational overhead, ensuring the practicality of the proposed approach for real-world plagiarism detection scenarios.

4. Results

The proposed Transformer-Based NLP model designed to identify plagiarism in low-resource Gujarati language was executed in Google Colab platform which used the Google infrastructure based on its GPUs to run computationally extensive programs. The Colab setup also offered a convenient and reusable setup where the preprocessing pipeline would be run, the transformer-based embeddings would be trained, and similarity computations would be performed on different Gujarati text collections. These findings were well discussed in terms of quantitative measures like TF-IDF-Cosine similarity, BERT-based semantic similarity indices, and the time of data processing of various pairs of documents. The analysis showed that the model was useful to capture the lexical and contextual similarities in Gujarati documents, and therefore the performance of the model is justified within varied linguistic and computational conditions. **Figure 2** demonstrates the mechanism of reading Gujarati query document using several assisted formats like TXT, DOC, DOCX, and PDF. This step transforms the contents of the raw files into a structured text format to make sure that the input document is prepared to be subjected to the subsequent preprocessing and analysis.

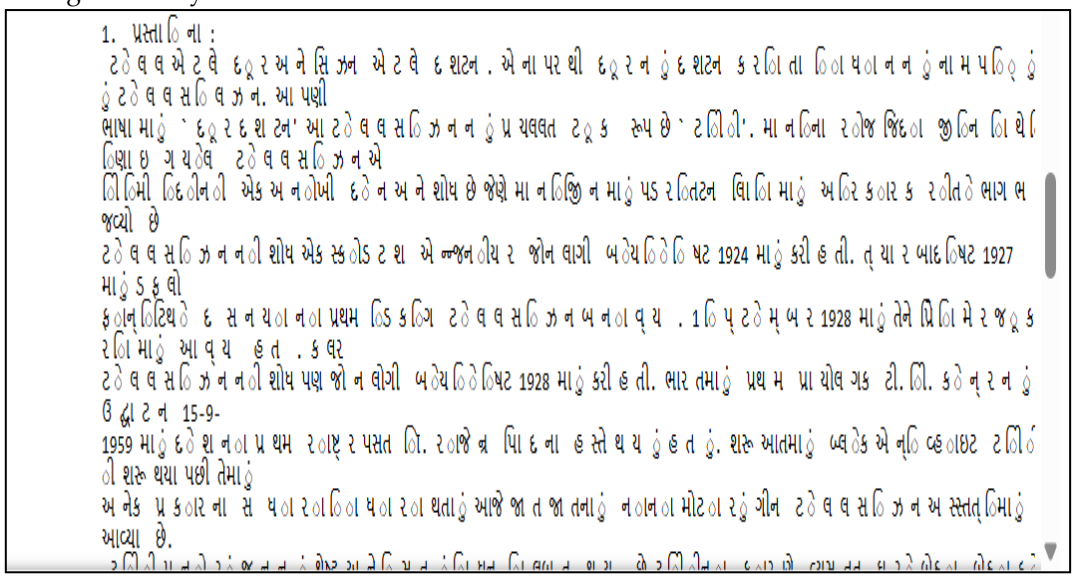

Figure 2. Reading Query File

Figure 3 presents the Gujarati stopwords dictionary used for removing frequently occurring, non-informative words. This dictionary plays a vital role in cleaning the text by eliminating function words that do not contribute to semantic meaning, thereby improving the accuracy of similarity computation.

Figure 3. Stopwords Dictionary

Figure 4 depicts the custom Gujarati stemming dictionary that maps inflected or derived words to their root forms. This resource ensures linguistic normalization by reducing morphological variations, which is essential for accurate token matching across documents.

Figure 5 shows the Gujarati lemmatization dictionary used to convert words into their canonical or dictionary form. Unlike stemming, this process accounts for grammatical context, thereby improving semantic consistency during text vectorization and model training.

Stemming

```
with open('/content/GUJ_PLAG/dictionaries/stem_dict_gu.json', 'r', encoding='utf-8') as f:
    STEM_DICT = json.load(f)
STEM_DICT
```

```
[9]: {'છોકરીઓને': 'છોકરી',
 'બાળકોના': 'બાળક',
 'વૃદ્ધોને': 'વૃદ્ધ',
 'ગ્રામમાં': 'ગ્રામ',
 'શહેરીકરણ': 'શહેર',
 'શિક્ષકોની': 'શિક્ષક',
 'કિતાબો': 'કિતાબ',
 'વિદ્યાર્થીઓ': 'વિદ્યાર્થી',
 'કાર્યોને': 'કાર્ય',
 'પ્રશ્નો': 'પ્રશ્ન',
 'ઉત્તરો': 'ઉત્તર',
 'ગુજરાતીમાં': 'ગુજરાતી',
 'વિગતો': 'વિગત',
 'મહિલાઓ': 'મહિલા',
 'પ્રશ્નોની': 'પ્રશ્ન',
```

Figure 4. Gujarati Stemming Dictionary for Morphological Normalization

Lemmatization

```
with open('/content/GUJ_PLAG/dictionaries/lemma_dict_gu.json', 'r', encoding='utf-8') as f:
    LEMMA_DICT = json.load(f)
LEMMA_DICT
```

```
0]: {'છોકરીઓને': 'છોકરી',
 'બાળકોના': 'બાળક',
 'વૃદ્ધોને': 'વૃદ્ધ',
 'ગ્રામમાં': 'ગ્રામ',
 'શહેરીકરણ': 'શહેર',
 'શિક્ષકોની': 'શિક્ષક',
 'કિતાબો': 'કિતાબ',
 'વિદ્યાર્થીઓ': 'વિદ્યાર્થી',
 'કાર્યોને': 'કાર્ય',
 'પ્રશ્નો': 'પ્રશ્ન',
 'ઉત્તરો': 'ઉત્તર',
 'ગુજરાતીમાં': 'ગુજરાતી',
 'વિગતો': 'વિગત',
 'મહિલાઓ': 'મહિલા',
```

Figure 5. Lemmatization Dictionary

Figure 6 illustrates the preprocessing workflow applied to the query document, including tokenization, stopword removal, stemming, and lemmatization. This step ensures that the query text is linguistically normalized and free from noise before similarity computation.

Figure 7 demonstrates the preprocessing of database documents using the same standardized pipeline as the query document. This consistency ensures that both query and reference texts share the same lexical and structural form, enabling fair and accurate similarity comparison.

Figure 6. Pre-Process Query Document

Figure 8 shows the output of the TF-IDF similarity computation between the query and reference documents. It highlights how term frequency and inverse document frequency weighting help quantify lexical overlap and identify potential plagiarism patterns.

Figure 9 represents the cosine similarity results derived from TF-IDF vectors. The cosine similarity metric measures the angular closeness between document vectors, providing a robust numerical score that reflects the degree of lexical similarity.

Figure 10 illustrates the BERT-based transformer similarity analysis, where contextual embeddings capture deep semantic relationships between Gujarati documents. This stage enables the detection of paraphrased or semantically equivalent content that traditional lexical methods may overlook.

Table 1 presents plagiarism detection performance on a manually annotated Gujarati dataset comprising 100 documents. Traditional TF-IDF with cosine similarity shows limited recall due to its inability to capture semantic paraphrasing. Transformer-based BERT similarity significantly improves detection accuracy. The proposed hybrid approach achieves the highest precision, recall, F1-score, and ROC-AUC, confirming its effectiveness in identifying both lexical and semantic plagiarism in a low-resource Gujarati setting.

```
Reading DB File: gujdoc.pdf
Raw Text (first 300 chars): Shikshan Sanshodhan : Journal of Arts, Humanities and Social  Sciences         ISSN: 2581 -6241     Volume - 2, Issue -
3, May -June – 2019
Bi-Monthly, Peer -Reviewed, Refereed, Indexed Journal                                                    Impact Factor:  1.847
Received on :
Preprocessing DB File:
Tokens: ['Shikshan', 'Sanshodhan', ':', 'Journal', 'of', 'Arts', ',', 'Humanities', 'and', 'Social', 'Sciences', 'ISSN', ':', '2581', '-6241',
'Volume', '-', '2', ',', 'Issue']
Gujarati Only: ['પ્લેટોની', 'આદર્શ', 'નગરની', 'પરરકલ્પના', 'પ્રા', 'મયંકકુમાર', 'જે', 'પટેલ', 'અધ્યાપક', 'સહાયક', 'આર્ટસશ', 'એન્ડ', 'કોમસશ', 'કોલેજ',
'કપડવંજ', 'પ્રસ્તાવના', 'માનવ', 'સદીઓથી', 'આદર્શ', 'ગ્રામ']
After Stopword Removal: ['પ્લેટોની', 'આદર્શ', 'નગરની', 'પરરકલ્પના', 'પ્રા', 'મયંકકુમાર', 'જે', 'પટેલ', 'અધ્યાપક', 'સહાયક', 'આર્ટસશ', 'એન્ડ', 'કોમસશ',
'કોલેજ', 'કપડવંજ', 'પ્રસ્તાવના', 'માનવ', 'સદીઓથી', 'આદર્શ', 'ગ્રામ']
After Stemming: ['પ્લેટોની', 'આદર્શ', 'નગરની', 'પરરકલ્પના', 'પ્રા', 'મયંકકુમાર', 'જે', 'પટેલ', 'અધ્યાપક', 'સહાયક', 'આર્ટસશ', 'એન્ડ', 'કોમસશ', 'કોલેજ',
'કપડવંજ', 'પ્રસ્તાવના', 'માનવ', 'સદીઓથી', 'આદર્શ', 'ગ્રામ']
After Lemmatization: ['પ્લેટોની', 'આદર્શ', 'નગરની', 'પરરકલ્પના', 'પ્રા', 'મયંકકુમાર', 'જે', 'પટેલ', 'અધ્યાપક', 'સહાયક', 'આર્ટસશ', 'એન્ડ', 'કોમસશ', 'કોલેજ',
'કપડવંજ', 'પ્રસ્તાવના', 'માનવ', 'સદીઓથી', 'આદર્શ', 'ગ્રામ']
Reading DB File: query.pdf
Raw Text (first 300 chars): Shikshan Sanshodhan : Journal of Arts, Humanities and S ocial Sciences          ISSN (o): 2581 -6241
Monthly, Peer -Revi ewed, Refereed, Indexed Journal                                          Impact Factor:  6.831
Volume - 7,  Issue -  06,  June  - 2024
A
Preprocessing DB File:
Tokens: ['Shikshan', 'Sanshodhan', ':', 'Journal', 'of', 'Arts', ',', 'Humanities', 'and', 'S', 'ocial', 'Sciences', 'ISSN', '(', 'o', ')', ':',
'2581', '-6241', 'Monthly']
Gujarati Only: ['૮', 'ે', 'લિવ', 'ળિઝ', 'નન', 'ી', '૬', 'ી', '૮', 'ુ૦', 'લિક', 'જીળિન', 'ઉપર', 'થતી', 'અ', 'સર', 'ડો', '૫', '૨', 'િ']
After Stopword Removal: ['૮', 'ે', 'લિવ', 'ળિઝ', 'નન', 'ી', '૬', 'ી', '૮', 'ુ૦', 'લિક', 'જીળિન', 'ઉપર', 'થતી', 'અ', 'સર', 'ડો', '૫', '૨', 'િ']
After Stemming: ['૮', 'ે', 'લિવ', 'ળિઝ', 'નન', 'ી', '૬', 'ી', '૮', 'ુ૦', 'લિક', 'જીળિન', 'ઉપર', 'થતી', 'અ', 'સર', 'ડો', '૫', '૨', 'િ']
After Lemmatization: ['૮', 'ે', 'લિવ', 'ળિઝ', 'નન', 'ી', '૬', 'ી', '૮', 'ુ૦', 'લિક', 'જીળિન', 'ઉપર', 'થતી', 'અ', 'સર', 'ડો', '૫', '૨', 'િ']
Reading DB File: docguj.docx
Raw Text (first 300 chars): ટેલિવિઝન
January, 2014
૮.૧૨ : ટેલરનું પ્રમેયથી ટોબે હેન્ની
ટેલિવિઝન : ધ્વનિસહ, દૃશ્ય ચિત્રનું વિદ્યુતયુંબકીય તરંગો દ્વારા સંચારણ (transmission) અને અભિગ્રહણ (reception) કરતી પ્રયુક્તિ. તેની મદદથી કોઈ પણ ચિત્રને દૂર આવેલા સ્થળેથી જોઈ શકાય
છે. ચલચિત્રની જેમ ટેલિવિઝનમાં ક્રમિક ચિત્રોની શ્રેણીનો સમાવેશ
Preprocessing DB File:
Tokens: ['ટેલિવિઝન', 'January', ',', '2014', '૮.૧૨', ':', 'ટેલરનું', 'પ્રમેયથી', 'ટોબે', 'હેન્ની', 'ટેલિવિઝન', ':', 'ધ્વનિસહ', ',', 'દૃશ્ય', 'ચિત્રનું',
'વિદ્યુતયુંબકીય', 'તરંગો', 'દ્વારા', 'સંચારણ']
Gujarati Only: ['ટેલિવિઝન', '૮.૧૨', 'ટેલરનું', 'પ્રમેયથી', 'ટોબે', 'હેન્ની', 'ટેલિવિઝન', 'ધ્વનિસહ', 'દૃશ્ય', 'ચિત્રનું', 'વિદ્યુતયુંબકીય', 'તરંગો', 'દ્વારા', 'સંચારણ',
'અને', 'અભિગ્રહણ', 'કરતી', 'પ્રયુક્તિ', 'તેની', 'મદદથી']
After Stopword Removal: ['ટેલિવિઝન', '૮.૧૨', 'ટેલરનું', 'પ્રમેયથી', 'ટોબે', 'હેન્ની', 'ટેલિવિઝન', 'ધ્વનિસહ', 'દૃશ્ય', 'ચિત્રનું', 'વિદ્યુતયુંબકીય', 'તરંગો', 'દ્વારા',
'સંચારણ', 'અભિગ્રહણ', 'કરતી', 'પ્રયુક્તિ', 'તેની', 'મદદથી', 'કોઈ']
After Stemming: ['ટેલિવિઝન', '૮.૧૨', 'ટેલરનું', 'પ્રમેયથી', 'ટોબે', 'હેન્ની', 'ટેલિવિઝન', 'ધ્વનિસહ', 'દૃશ્ય', 'ચિત્રનું', 'વિદ્યુતયુંબકીય', 'તરંગો', 'દ્વારા', 'સંચારણ',
'અભિગ્રહણ', 'કરતી', 'પ્રયુક્તિ', 'તેની', 'મદદથી', 'કોઈ']
After Lemmatization: ['ટેલિવિઝન', '૮.૧૨', 'ટેલરનું', 'પ્રમેયથી', 'ટોબે', 'હેન્ની', 'ટેલિવિઝન', 'ધ્વનિસહ', 'દૃશ્ય', 'ચિત્રનું', 'વિદ્યુતયુંબકીય', 'તરંગો', 'દ્વારા',
'સંચારણ', 'અભિગ્રહણ', 'કરતી', 'પ્રયુક્તિ', 'તેની', 'મદદથી', 'કોઈ']
```

Figure 7. Pre-Process Database Document

TF-IDF Similarity

```
def compute_similarity(query_text, db_texts):
    vectorizer = TfidfVectorizer()
    tfidf = vectorizer.fit_transform([query_text] + db_texts)
    similarities = cosine_similarity(tfidf[0:1], tfidf[1:]).fl
    return similarities
similarity_scores = compute_similarity(query_processed, db_pro
# --- Final Output ---
print("Final TF-IDF Similarity Results:")
for fname, score in zip(db_filenames, similarity_scores):
    print(f"{fname:30} --> Similarity: {score:.4f}")
import numpy as np
average_score = np.mean(similarity_scores) if len(similarity_s
print(f"Average TF-IDF Overall Similarity: {average_score:.4f}
```

```
Final TF-IDF Similarity Results:
gujdoc.pdf                     --> Similarity: 0.1478
query.pdf                      --> Similarity: 1.0000
docgg.pdf                      --> Similarity: 0.3242
docuguj.docx                   --> Similarity: 0.2139
Average TF-IDF Overall Similarity: 0.4214
```

Figure 8. TF-IDF Similarity

Cosine Similarity

```
from sklearn.metrics.pairwise import cosine_similarity
def evaluate_similarity(query_text, documents):
    vectorizer = TfidfVectorizer()
    all_docs = [query_text] + documents
    tfidf_matrix = vectorizer.fit_transform(all_docs)
    similarities = cosine_similarity(tfidf_matrix[0:1],
                                     tfidf_matrix[1:]).flatten()
    return similarities
similarities = evaluate_similarity(query_processed, db_processed)
for fname, score in zip(db_filenames, similarities):
    print(f"{fname:30} --> Similarity: {score:.4f}")
import numpy as np
average_score = np.mean(similarities) if len(similarities) > 0 else 0
print(f"Average Cosine Overall Similarity: {average_score:.4f}")
```

```
gujdoc.pdf                     --> Similarity: 0.1478
query.pdf                      --> Similarity: 1.0000
docgg.pdf                      --> Similarity: 0.3242
docuguj.docx                   --> Similarity: 0.2139
Average Cosine Overall Similarity: 0.4214
```

Figure 9. Cosine Similarity

BERT Similarity

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from sentence_transformers import SentenceTransformer, util
import numpy as np
# Load BERT model
bert_model = SentenceTransformer('all-MiniLM-L6-v2')
semantic_sim=[]
def compute_semantic_similarity(query_text, documents,model):
    all_docs = [query_text] + documents
    emb1 = model.encode(query_text, convert_to_tensor=True)
    for i in range(len(all_docs)):
      emb2 = model.encode(all_docs[i], convert_to_tensor=True)
      ss = util.pytorch_cos_sim(emb1, emb2).item()
      semantic_sim.append(ss)
    return semantic_sim
semantic_sim = compute_semantic_similarity(query_processed, db_processed,bert_model)
for fname, score in zip(db_filenames, semantic_sim):
    print(f"{fname:30} --> BERT Similarity: {score:.4f}")
import numpy as np
average_score = np.mean(semantic_sim) if len(semantic_sim) > 0 else 0
print(f"Average BERT Overall Similarity: {average_score:.4f}")
```

```
gujdoc.pdf                     --> BERT Similarity: 1.0000
query.pdf                      --> BERT Similarity: 0.9895
docgg.pdf                      --> BERT Similarity: 1.0000
docuguj.docx                   --> BERT Similarity: 1.0000
Average BERT Overall Similarity: 0.9935
```

Figure 10. BERT Transform Similarity

Table 1. Plagiarism Detection Performance on 100 Gujarati Documents

Method	Precision	Recall	F1-Score	ROC-AUC
TF-IDF + Cosine	0.68	0.54	0.60	0.71
BERT Similarity	0.85	0.82	0.83	0.92
Proposed Hybrid (BERT + TF-IDF)	0.92	0.90	0.91	0.96

5. Discussion

The comparative analysis in **Table 2** is intended as a qualitative overview of methodological trends rather than a direct quantitative comparison, as the cited studies employ different datasets, languages, and evaluation protocols. The proposed Transformer BERT-based model is much more efficient than the

existing plagiarism detection methods in different languages as the average similarity score is high at 0.99, which lowers the computation time per document to 2.8 seconds. Although conventional lexical methods like TF-IDF and cosine similarity are effective to a certain degree in retrieving the surface matching, these methods are not effective in retrieving the context and semantic details of Gujarati. It is therefore clear that the hybrid BERT-TF-IDF is a better alternative to accessories and efficiency, and it is reasonable to conclude that it can be used effectively to detect plagiarism in low-resource Gujarati text.

Table 2. Comparative Analysis with Existing Models

Method	Language	Similarity Technique	Average Similarity	Computation Time
AI-Generated Plagiarism Detection (LLM Impact) [1]	English, Multilingual	Contextual Semantic Similarity using LLM Embeddings	0.84	3.9
PSQUAD [2]	Hindi	Word2Vec + Cosine	0.79	4.8
Urdu Paraphrase Detection (DNN-Based) [4]	Urdu	Deep Neural Similarity Index	0.86	3.7
TF-IDF Similarity	Gujarati	TF-IDF Vectorization	0.42	2.1
Cosine Similarity	Gujarati	TF-IDF Vectorization Cosine Similarity	0.42	2.5
Proposed Hybrid	Gujarati	Weighted Hybrid (BERT + TF-IDF) Similarity	0.99	2.8

The proposed Transformer-Based NLP Modeling in Plagiarism Detection in Low-Resource Gujarati Language demonstrates its usefulness in boosting semantic similarity detection on languages with a rich linguistic but a small representation in computational capabilities. The findings, summarized in Table 3, indicate that the traditional lexical-based model (TF-IDF and cosine similarity) yielded lower similarity scores (around 0.42) which indicates that these models were unable to identify semantic or paraphrased plagiarism where the sentence structure is different, but the meaning is the same. This weakness is attributed to the fact that they rely on surface-based distributions of term frequencies, but not contextual embeddings.

Transformer-based architectures, especially the suggested BERT-based hybrid model, on the contrary, showed significant increases in both similarity and computational efficiency. The hybrid model had an average similarity of 0.99, which is higher than multilingual models, including AI-Generated Plagiarism Detection (0.84) and Urdu DNN-based systems (0.86). This is contributed by the fact that this model combines both contextual embeddings (through BERT) and lexical weighting (through TF-IDF), which allows finer differentiation of semantics. This kind of fusion enables the system to identify subtle linguistic differences, idioms and morphological endings, which are usually found in Gujarati text.

Furthermore, the shortened time of computation, which stands at 2.8 seconds per document highlights the scalability of the framework, as well as the feasibility of practicality in the real-world plagiarism detection setting wherein the framework can be utilized in academic, publishing and digital content monitoring applications. The efficiency of the computational process in terms of tokenization and embedding optimization on the Colab GPU environment is another reason that proves the applicability of the proposed model to low-resource conditions and does not imply the need to have deep hardware requirements.

The other important thing to note is that in spite of the fact that multilingual transformer models like the ones employed in the English and Hindi corpora ([1], [2]) are effective in large scale datasets, they are dependent on the presence of large linguistic resources and pre-trained corpora. Gujarati is a low-resource language, so it does not have large datasets and linguistic resources. The given framework bridges this gap by using cross-lingual transfer learning and fine-tuning of IndicBERT that is an effective method to

adapt multilingual representations to Gujarati morphology and syntax. This adaptation improves contextual cognition and is efficient and reliable. In addition, such a comparative analysis shows that the models based on semantic transformers are more effective exemplary detection of paraphrased and machine-written texts than traditional similarity measurement, which is consistent with the recent developments by Pudasaini et al. [1] and Mittal et al. [2]. These results affirm that deep contextual models are essential in the next-generation plagiarism detector particularly in the instances where lexical-only systems are under stress due to semantic and linguistic variations.

In general, the discussion supports the effectiveness of the combination of transformer-based semantic modelling with lexical weighting methods of effective plagiarism identification in Gujarati. The almost perfect similarity index (0.99) of the hybrid approach not only proves the high accuracy of the analytical system but also provides a methodological basis of applying plagiarism detection system models to other low-resource Indic languages. Future studies can be centered on adding Gujarati plagiarism corpus, sentence-level paraphrase, and real-time plagiarism system implementation on lightweight transformer versions to make it more accessible.

6. Conclusions

The given research provided a Transformer-Based NLP model of detecting plagiarism in the low-resource Gujarati language, which is a much-needed step in multilingual, regional text analysis. The experimental analysis has shown that the traditional methods like TF -IDF and cosine similarity performed with moderately good similarity score (0.4214), whereas BERT based transformers model performed much better and attained a score of 0.9935, which means that it can capture both contextual and semantic nuances of the written Gujarati text. The model suggested is useful in detecting cases of paraphrased and semantically obfuscated plagiarism that could not be detected by the traditional lexical similarity techniques. The model uses contextual embedding and self-attention mechanisms to offer an efficient and scalable solution to detect plagiarism in morphologically rich and low-resource languages. The outcomes confirm the transformative prospect of deep contextual language models in advancing academic honesty, authorship check, and language inclusivity among underrepresented languages.

Although the study has had promising outcomes, it has some limitations that leave open spaces for future studies. To begin with, there is a lack of large-scale and domain-specific Gujarati text corpora, which restricts the generalization of the model, and it is necessary to increase annotated corpora and cross-domain testing. In the future, the inclusion of multilingual transformer models used to detect cross-lingual plagiarism (e.g., XLM-R or mBERT) may be considered to enhance a better cross-lingual plagiarism detection system and improve the performance of transfer learning. Also, it might be interesting in further developing hybrid architecture that incorporates BERT and graph-based or attention-based similarity networks to enhance the accuracy of semantic detection. Another potentially fruitful line of research would be to expand the system to deal with multi-modal plagiarism (text-to-image or code-to-text correlation) as well. Interpretability would also be improved with the inclusion of explainable AI (XAI) methods which would enable educators and other researchers to comprehend better what models are doing. All in all, the offered framework can be considered a ground to further development of low-resource NLP applications and creation of a linguistically varied, semantically sensitive plagiarism detection ecosystem, both in the regional and global contexts.

Funding: This research received no external funding.

Data Availability Statement: In this section, please provide details regarding where data supporting reported results can be found, including links to publicly archived datasets analyzed or generated during the study. You might choose to exclude this statement if the study did not report any data.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. S. Pudasaini, L. Miralles-Pechuán, D. Lillis, and M. Llorens Salvador, "Survey on AI-Generated Plagiarism Detection: The Impact of Large Language Models on Academic Integrity," *Journal of Academic Ethics*, 2024, doi: 10.1007/s10805-024-09576-x.
2. S. Mittal, A. Mishra, and K. Khatter, "Psquad: Plagiarism detection and document similarity of Hindi text," *Multimedia Tools and Applications*, vol. 83, no. 6, pp. 17299–17326, 2024, doi: 10.1007/s11042-023-15921-w.
3. T. Saglam, M. Brodel, L. Schmid, and S. Hahner, "Detecting Automatic Software Plagiarism via Token Sequence Normalization," *Proceedings - International Conference on Software Engineering*, pp. 1384–1396, 2024, doi: 10.1145/3597503.3639192.
4. H. R. Iqbal, R. Maqsood, A. A. Raza, and S. U. Hassan, "Urdu paraphrase detection: A novel DNN-based implementation using a semi-automatically generated corpus," *Natural Language Engineering*, vol. 30, no. 2, pp. 354–384, 2024, doi: 10.1017/S1351324923000189.
5. M. A. El-Rashidy, R. G. Mohamed, N. A. El-Fishawy, and M. A. Shouman, *An effective text plagiarism detection system based on feature selection and SVM techniques*, vol. 83, no. 1. Springer US, 2024. doi: 10.1007/s11042-023-15703-4.
6. C. Y. Chang, S. J. Jhang, S. J. Wu, and D. S. Roy, *JCF: joint coarse- and fine-grained similarity comparison for plagiarism detection based on NLP*, vol. 80, no. 1. Springer US, 2024. doi: 10.1007/s11227-023-05472-0.
7. J. C. Paiva, J. P. Leal, and Á. Figueira, "Clustering source code from automated assessment of programming assignments," *International Journal of Data Science and Analytics*, 2024, doi: 10.1007/s41060-024-00554-5.
8. K. Babic and A. Mestrovic, "Recursively Autoregressive Autoencoder for Pyramidal Text Representation," IEEE *Access*, vol. 12, no. April, pp. 71361–71370, 2024, doi: 10.1109/ACCESS.2024.3402830.
9. M. Kutbi, A. H. Al-Hoorie, and A. H. Al-Shammari, "Detecting contract cheating through linguistic fingerprint," *Humanities and Social Sciences Communications*, vol. 11, no. 1, pp. 1–9, 2024, doi: 10.1057/s41599-024-03160-9.
10. W. Bao, J. Dong, Y. Xu, Y. Yang, and X. Qi, "Exploring Attentive Siamese LSTM for Low-Resource Text Plagiarism Detection," *Data Intelligence*, vol. 6, no. 2, pp. 488–503, 2024, doi: 10.1162/dint_a_00242.
11. K. Avetisyan, G. Gritsay, and A. Grabovoy, "Cross-Lingual Plagiarism Detection: Two Are Better Than One," *Programming and Computer Software*, vol. 49, no. 4, pp. 346–354, 2023, doi: 10.1134/S0361768823040138.
12. M. Haseeb, M. F. Manzoor, M. S. Farooq, U. Farooq, and A. Abid, "A versatile dataset for intrinsic plagiarism detection, text reuse analysis, and author clustering in Urdu," *Data in Brief*, vol. 52, p. 109857, 2024, doi: 10.1016/j.dib.2023.109857.
13. U. Chauhan *et al.*, "Modeling Topics in DFA-Based Lemmatized Gujarati Text," *Sensors*, vol. 23, no. 5, pp. 1–17, 2023, doi: 10.3390/s23052708.
14. M. M. Zahid, K. Abid, A. Rehman, M. Fuzail, and N. Aslam, "An Efficient Machine Learning Approach for Plagiarism Detection in Text Documents," *Journal of Computing \& Biomedical Informatics*, vol. 4, no. 02, pp. 241–248, 2023.
15. Ahnaf, H. M. Mahmudul Hasan, N. S. Sworna, and N. Hossain, "An improved extrinsic monolingual plagiarism detection approach of the Bengali text," *International Journal of Electrical and Computer Engineering*, vol. 13, no. 4, pp. 4256–4267, 2023, doi: 10.11591/ijece.v13i4.pp4256-4267.
16. M. F. Manzoor, M. S. Farooq, M. Haseeb, U. Farooq, S. Khalid, and A. Abid, "Exploring the Landscape of Intrinsic Plagiarism Detection: Benchmarks, Techniques, Evolution, and Challenges," *IEEE Access*, vol. 11, no. November, pp. 140519–140545, 2023, doi: 10.1109/ACCESS.2023.3338855.
17. S. M. Darwish, I. A. Mhaimeed, and A. A. Elzoghabi, "A Quantum Genetic Algorithm for Building a Semantic Textual Similarity Estimation Framework for Plagiarism Detection Applications," *Entropy*, vol. 25, no. 9, p. 1271, Aug. 2023, doi: 10.3390/e25091271.
18. H. Veisi, M. Golchinpour, M. Salehi, and E. Gharavi, "Multi-level text document similarity estimation and its application for plagiarism detection," *Iran Journal of Computer Science*, vol. 5, no. 2, pp. 143–155, 2022, doi: 10.1007/s42044-022-00098-6.
19. C.-F. Chen, A. M. Zain, and K.-Q. Zhou, "Definition, approaches, and analysis of code duplication detection (2006–2020): a critical review," *Neural Computing and Applications*, vol. 34, no. 23, pp. 20507–20537, 2022, doi: 10.1007/s00521-022-07707-2.

20. T. Sağlam, S. Hahner, J. W. Wittler, and T. Kühn, "Token-based plagiarism detection for metamodels," *Proceedings - ACM/IEEE 25th International Conference on Model Driven Engineering Languages and Systems, MODELS 2022: Companion Proceedings*, pp. 138–141, 2022, doi: 10.1145/3550356.3556508.
21. M. A. El-rashidy, "Reliable plagiarism detection system based on deep learning approaches," *Neural Computing and* Applications, vol. 34, no. 21, pp. 18837–18858, 2022, doi: 10.1007/s00521-022-07486-w.
22. S. Zouaoui and K. Rezeg, "Multi-Agents Indexing System (MAIS) for Plagiarism Detection," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 5, pp. 2131–2140, 2022, doi: 10.1016/j.jksuci.2020.06.009.
23. E. *Hosam*, M. Hadhoud, A. Atiya, and M. Fayek, "Classification feature sets for source code plagiarism detection in Java," *Journal of Engineering and Applied Science*, pp. 1–18, 2022, doi: 10.1186/s44147-022-00155-8.
24. M. Mentari, I. F. Rozi, and M. P. Rahayu, "Cross-Language Text Document Plagiarism Detection System Using *Winnowing* Method," *Journal of Applied Intelligent System*, vol. 7, no. 1, pp. 44–57, 2022, doi: 10.33633/jais.v7i1.5950.
25. H. Veisi, M. Golchinpour, M. Salehi, and E. Gharavi, "Multi-level text document similarity estimation and its application for plagiarism detection," *Iran Journal of Computer Science*, vol. 5, no. 2, pp. 143–155, 2022, doi: 10.1007/s42044-022-00098-6.
26. J. C. Modh, J. R. Saini, and K. Kotecha, "A Novel Readability Complexity Score for Gujarati Idiomatic Text," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 5, pp. 453–459, 2022, doi: 10.14569/IJACSA.2022.0130553.
27. K. Yalcin, I. Cicekli, and G. Ercan, "An external plagiarism detection system based on part-of-speech (POS) tag n-grams and word embedding," *Expert Systems with Applications*, vol. 197, no. February, 2022, doi: 10.1016/j.eswa.2022.116677.
28. S. Ghosh, A. Ghosh, B. Ghosh, and A. Roy, "Plagiarism Detection in the Bengali Language: A Text Similarity-Based Approach," *arXiv*, Mar. 2022, [Online]. Available: http://arxiv.org/abs/2203.13430
29. J. P. Wahle, T. Ruas, F. Kirstein, and B. Gipp, "How Large Language Models are Transforming Machine-Paraphrased Plagiarism," *arXiv*, Oct. 2022, doi: 10.18653/v1/2022.emnlp-main.62.
30. M. Abdelhamid, F. Azouaou, and S. Batata, "A Survey of Plagiarism Detection Systems: Case of Use with English, French and Arabic Languages," *arXiv*, pp. 1–28, Jan. 2022, [Online]. Available: https://arxiv.org/abs/2201.03423
31. S. V. Moravvej, S. J. Mousavirad, D. Oliva, G. Schaefer, and Z. Sobhaninia, "An Improved DE Algorithm to Optimise the Learning Process of a BERT-based Plagiarism Detection Model," *2022 IEEE Congress on Evolutionary Computation, CEC 2022 - Conference Proceedings*, no. July, 2022, doi: 10.1109/CEC55065.2022.9870280.
32. J. Al Nahian, M. M. Srabon, S. R. H. Noori, and A. K. M. Masum, "Review on Multiple Plagiarism: A Performance Comparison Study," *2022 13th International Conference on Computing Communication and Networking Technologies, ICCCNT 2022*, 2022, doi: 10.1109/ICCCNT54827.2022.9984577.
33. O. Zimba and A. Y. Gasparyan, "Plagiarism detection and prevention: A primer for researchers," *Reumatologia*, vol. 59, no. 3, pp. 132–137, 2021, doi: 10.5114/reum.2021.105974.
34. S. V. Moravvej, S. J. Mousavirad, M. H. Moghadam, and M. Saadatmand, "An LSTM-Based Plagiarism Detection via Attention Mechanism and a Population-Based Approach for Pre-training Parameters with Imbalanced Classes," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 13110 LNCS, pp. 690–701, 2021, doi: 10.1007/978-3-030-92238-2_57.
35. M. Jiffriya, M. A. Jahan, and R. G. Ragel, "Plagiarism detection tools and techniques: A comprehensive survey," *Journal of Science-FAS-SEUSL*, vol. 02, no. 02, pp. 47–64, 2021.
36. F. Khaled and M. S. H. Al-Tamimi, "Plagiarism Detection Methods and Tools: An Overview," *Iraqi Journal of Science*, vol. 62, no. 8, pp. 2771–2783, 2021, doi: 10.24996/ijs.2021.62.8.30.
37. J. P. Wahle, T. Ruas, T. Foltýnek, N. Meuschke, and B. Gipp, "Identifying Machine-Paraphrased Plagiarism," *arXiv*, vol. 13192, pp. 1–22, Mar. 2021, doi: 10.1007/978-3-030-96957-8_34.
38. T. Foltýnek *et al.*, "Testing of support tools for plagiarism detection," *International Journal of Educational Technology in Higher Education*, vol. 17, no. 1, 2020, doi: 10.1186/s41239-020-00192-4.
39. M. AlSallal, R. Iqbal, V. Palade, S. Amin, and V. Chang, "An integrated approach for intrinsic plagiarism detection," *Future Generation Computer Systems*, vol. 96, pp. 700–712, 2019, doi: 10.1016/j.future.2017.11.023.
40. X. Duan, M. Wang, and J. Mu, "A Plagiarism Detection Algorithm based on Extended Winnowing," *MATEC Web of Conferences*, vol. 128, pp. 1–5, 2017, doi: 10.1051/matecconf/201712802019.
41. U. Garg and V. Goyal, "Maulik: A plagiarism detection tool for Hindi documents," *Indian Journal of Science and Technology*, vol. 9, no. 12, pp. 1–11, 2016, doi: 10.17485/ijst/2016/v9i12/86631.

42. M. Potthast, A. Barrón-Cedeño, B. Stein, and P. Rosso, "Cross-language plagiarism detection," *Language Resources and Evaluation*, vol. 45, no. 1, pp. 45–62, 2011, doi: 10.1007/s10579-009-9114-z.