

DECENTRALIZED INTELLECTUAL PROPERTY (IP) PROTECTION PLATFORM WITH AI-POWERED SIMILARITY DETECTION

Project Id: R25-016

Project Proposal Report

Wanigasuriya W.P.S.D

B.Sc. (Hons) in Information Technology Specialized in Software Engineering

Department of Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

February 2025

**SIMILARITY CHECKING FOR TEXT BASE CONTENTS
(BOOKS / NOVELS) AND CONTINUOUS MODEL LEARNING**

Project Id: R25-016

Project Proposal Report

Wanigasuriya W.P.S.D

B.Sc. (Hons) in Information Technology Specialized in Software Engineering

Department of Software Engineering


Sri Lanka Institute of Information Technology

Sri Lanka

February 2025


A DECLARATION, COPYRIGHT STATEMENT, AND THE STATEMENT OF THE SUPERVISOR.

We declare that this is our own work, and this proposal does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any other university or institute of higher learning, and to the best of our knowledge and belief, it does not contain any material previously published or written by another person except where the acknowledgment is made in the text.

Name	Student ID	Signature
Wanigasuriya W.P.S. D	IT21814242	

Signature of the Supervisor
(Dr. Dharshana Kasthurirathna)

Date


.....

2025/02/02
.....

Signature of the Co-Supervisor
(Dr. Kalpani Manathunga)

Date

.....

.....

ABSTRACT

The rapid digitization of literary works, including novels, books, and research articles, has raised concerns regarding intellectual property (IP) infringement and plagiarism. Traditional plagiarism detection tools such as Turnitin, Grammarly, and Copyscape offer basic text-matching capabilities but fail to detect deep semantic similarities, paraphrased content, multilingual plagiarism, and overlaps in storyline or characters. The core research problem is the inability of existing tools to analyze creative literary works beyond surface-level textual matching.

This study proposes an AI-driven similarity-checking system to address these challenges by leveraging advanced Natural Language Processing (NLP) and Machine Learning (ML) techniques. The system integrates transformer-based models such as BERT, RoBERTa, Pegasus, and XLM-R to detect reworded content, paraphrased ideas, and translated plagiarism. Additionally, graph-based models using Neo4j and NetworkX will analyze storyline structures, character relationships, and thematic similarities to provide a comprehensive approach to plagiarism detection.

The research follows a structured methodology, including data collection from literary sources, text pre-processing with NLP techniques, supervised learning for model training, and evaluation using benchmark datasets. Adaptive learning mechanisms will be incorporated to improve detection accuracy through real-time feedback and new data.

The expected outcomes include enhanced plagiarism detection accuracy through semantic analysis, cross-language similarity identification, and creative content protection through novel storyline and character analysis. A comprehensive reporting system will be developed for authors and publishers to safeguard intellectual property. This study will contribute to the development of a next-generation plagiarism detection tool for the digital publishing industry.

Keywords: NLP, AI/ML, Plagiarism Detection, Semantic Similarity, Transformer Models, Graph-Based Analysis, Adaptive Learning, Intellectual Property Protection

TABLE OF CONTENTS

A DECLARATION, COPYRIGHT STATEMENT, AND THE STATEMENT OF THE SUPERVISOR.	i
ABSTRACT.....	ii
TABLE OF CONTENTS.....	iii
LIST OF FIGURES	v
LIST OF TABLES.....	vi
LIST OF ABBREVIATIONS.....	vii
1. INTRODUCTION	1
1.1. Background & Literature Survey	1
1.2. Existing Plagiarism Detection Tools and Their Shortcomings.....	2
1.3. State-of-the-Art Research and AI-Based Solutions	3
1.4. Relevant Studies and Techniques Required	4
1.4.1. Deep Learning & NLP for Text Similarity Analysis	4
1.4.2. Multilingual NLP & Cross-Language Detection	4
1.4.3. Graph-Based Storyline and Character Similarity Detection	4
1.4.4. Adaptive Learning for Continuous Model Improvement.....	4
1.5. Research Gap.....	5
1.6. Research Problem.....	7
2. OBJECTIVES	9
2.1. Main Objective	9
2.2. Specific Objectives	9
3. METHODOLOGY	12
3.1. System Overview.....	12
3.2. System Architecture & Component Diagram.....	14
3.3. Tasks and Sub-Tasks for Implementation	15
3.4. Required Resources & Tools	17
3.5. Data Collection & Sources	18
3.6. Project Timeline & Task Schedule.....	18
3.7. Anticipated Outcomes & Real-World Applications	19
4. PROJECT REQUIREMENTS	20
4.1. Functional Requirements.....	20
4.1.1. Text Similarity Detection.....	20
4.1.2. Paraphrase Identification.....	20
4.1.3. Cross-Language Similarity Detection.....	21
4.1.4. Storyline & Character Similarity Analysis	21

4.1.5. Adaptive Learning Mechanism	21
4.1.6. Real-Time Plagiarism Reporting	21
4.2. Non-Functional Requirements.....	22
4.2.1. Performance Requirement.....	22
4.2.2. Scalability Requirements	22
4.2.3. Security & Data Privacy	22
4.2.4. Availability & Reliability.....	23
4.3. Technologies & Tools	23
4.4. User Requirements	24
4.5. Use Cases.....	25
4.6. Test Cases	26
4.7. Commercialization	27
4.7.1. Target Market.....	27
4.7.2. Revenue Model	27
4.7.3. Why Choose Our Solution?	27
5. DESCRIPTION OF PERSONAL AND FACILITIES	28
5.1. Undergraduate Researchers	28
5.2. Supervisory Team	28
6. REFERENCE LIST	29
6.1. Plagiarism Detection & Text Similarity Analysis	29
6.2. NLP-Based Paraphrase & Semantic Similarity Detection.....	29
6.3. Cross-Language Plagiarism Detection	30
6.4. Graph-Based Storyline & Character Similarity Analysis	30
6.5. Adaptive Learning & AI Model Improvement.....	30
6.6. Blockchain-Based IP Protection & Smart Contracts	31
6.7. Ethical AI & Legal Considerations	31
7. APPENDICES	32
7.1. Gantt chart	32
7.2. Work Breakdown Structure	32
7.3. Plagiarism Report	33

LIST OF FIGURES

Figure 1 System Overview Diagram..... 13

Figure 2 Component Diagram..... 14

Figure 3 Gantt Chart 32

Figure 4 Work Breakdown Structure (WBS)..... 32

Figure 5 Plagiarism Report 33

LIST OF TABLES

Table 1 List of Abbreviations	vii
Table 2 Research Gaps in Existing Tools	6
Table 3 Feature Comparison Between Existing Tools and Proposed System	8
Table 4 Project Timeline & Task Schedule	18
Table 5 Functional Requirements	22
Table 6 Non-Functional Requirements	23
Table 7 Technologies & Tools.....	24
Table 8 User Requirements.....	24
Table 9 Use Cases.....	25
Table 10 Test Cases	26
Table 11 Undergraduate Researchers	28
Table 12 Supervisory Team	28

LIST OF ABBREVIATIONS

Abbreviation	Full Form
IP	Intellectual Property
NLP	Natural Language Processing
AI	Artificial Intelligence
ML	Machine Learning
BERT	Bidirectional Encoder Representations from Transformers
RoBERTa	A Robustly Optimized BERT Pretraining Approach
mBERT	Multilingual BERT
XLNet	Cross-Lingual Language Model RoBERTa
T5	Text-to-Text Transfer Transformer
GNN	Graph Neural Networks
SBERT	Sentence-BERT
RLHF	Reinforcement Learning with Human Feedback
AWS	Amazon Web Services
EC2	Elastic Compute Cloud (AWS Service)
GDPR	General Data Protection Regulation
RBAC	Role-Based Access Control
ACL	Association for Computational Linguistics
ICML	International Conference on Machine Learning
NeurIPS	Neural Information Processing Systems
MIT	Massachusetts Institute of Technology
SQL	Structured Query Language
GPU	Graphics Processing Unit
API	Application Programming Interface
GloVe	Global Vectors for Word Representation
CNN	Convolutional Neural Network
LSTM	Long Short-Term Memory
GAN	Generative Adversarial Network
RL	Reinforcement Learning

Table 1 List of Abbreviations

1. INTRODUCTION

1.1. Background & Literature Survey

The study of text similarity detection has been an evolving area of research, driven by the increasing need to protect intellectual property (IP) in digital content. Research in this field has primarily focused on identifying textual similarities, plagiarism, and content duplication, particularly in academic writing, research publications, and creative literary works.

Early research in text similarity detection was based on string-matching algorithms, which compared documents based on exact word overlap and character sequences. Studies proposed methods such as Levenshtein distance, Jaccard similarity, and cosine similarity, which calculated the textual similarity by analyzing word-level matches. However, researchers quickly identified the limitations of these approaches, particularly in handling paraphrased, summarized, and translated text, which retained the same meaning but were expressed differently.

To overcome these challenges, researchers explored statistical and rule-based approaches for detecting similarity between texts. The vector space model (VSM) was introduced, representing documents as vectors in a multi-dimensional space to measure their similarity based on word frequency distributions. Another significant advancement came with the Latent Semantic Analysis (LSA) technique, which allowed researchers to capture conceptual relationships between words rather than relying on direct text matching.

With the advent of Natural Language Processing (NLP), research in text similarity detection took a major leap forward. Researchers began leveraging syntactic and semantic analysis techniques to identify deeper contextual similarities between texts. Studies explored dependency parsing, named entity recognition (NER), and part-of-speech (POS) tagging to enhance similarity detection accuracy. Additionally, the rise of machine learning (ML) algorithms enabled researchers to train models on large datasets, improving their ability to detect reworded or paraphrased content.

Recent research has introduced deep learning and transformer-based models, such as BERT (Bidirectional Encoder Representations from Transformers), which provide a context-aware approach to text similarity detection. These models have demonstrated superior performance in understanding the meaning and relationships between words, making them highly effective in identifying semantic similarities across different types of content.

Furthermore, research has expanded into multilingual text similarity detection, where studies

have investigated how cross-language plagiarism can be identified using multilingual embeddings and translation-aware models. Some studies have proposed the use of graph-based techniques to analyze narrative structures and character relationships in literary works, allowing for a more comprehensive analysis of content similarities beyond just text-level comparisons.

Overall, research in this domain has continuously evolved from basic text-matching algorithms to advanced NLP and AI-driven approaches, addressing the growing complexities of content originality verification, plagiarism detection, and intellectual property protection.

1.2. Existing Plagiarism Detection Tools and Their Shortcomings

Traditional plagiarism detection tools such as Turnitin, Grammarly, Copyscape, and Plagscan primarily rely on text-matching algorithms that compare documents based on exact word sequences and simple paraphrasing techniques. These tools are widely utilized in academic settings to detect plagiarism in research papers, essays, and reports. However, they are less effective when applied to creative literary works, where text can be rewritten, restructured, or translated while retaining the same core narrative, ideas, and character relationships.

The primary limitations of existing plagiarism detection systems include:

- ✚ **Surface-Level Text Matching:** These systems rely on n-gram comparisons and word overlap analysis, which fail to capture the semantic and contextual meaning [1].
- ✚ **Inability to Detect Paraphrased or Rewritten Content:** Many tools struggle with detecting deeply reworded or paraphrased passages [7].
- ✚ **No Cross-Language Similarity Checking:** Translations of literary works into different languages often go undetected [11][12].
- ✚ **Lack of Storyline or Character-Based Plagiarism Detection:** Traditional systems fail to account for creative elements such as plot structures, character arcs, and thematic elements, which can be copied and slightly modified without triggering plagiarism detection [15].

1.3. State-of-the-Art Research and AI-Based Solutions

Recent advancements in Artificial Intelligence (AI) and Natural Language Processing (NLP) have introduced transformer-based models that enable a deeper understanding of textual content. These include:

- ✚ BERT (Bidirectional Encoder Representations from Transformers): Captures contextual similarities in text [2].
- ✚ RoBERTa (A Robustly Optimized BERT Pretraining Approach): Enhances semantic similarity detection [2].
- ✚ mBERT & XLM-R (Multilingual BERT & Cross-lingual Language Model): Enable cross-language text analysis for detecting translated plagiarism [11][12].
- ✚ Pegasus & T5: Specialized models for paraphrase detection and text rephrasing [9].

Additionally, graph-based AI techniques such as **Neo4j** and **NetworkX** are now being applied to analyze storyline and character relationships, offering a novel approach to creative plagiarism detection [15].

1.4. Relevant Studies and Techniques Required

To develop an AI-powered similarity-checking system, several research areas and techniques must be mastered, including:

1.4.1. Deep Learning & NLP for Text Similarity Analysis

- ✚ Transformer-Based Language Models (BERT, RoBERTa, XLM-R): For semantic similarity detection.
- ✚ Sentence-BERT (SBERT): Converts text into high-dimensional embeddings to compare contextual similarities.
- ✚ Word Embeddings (Word2Vec, GloVe, FastText): Captures word relationships and synonyms for better paraphrase detection.

1.4.2. Multilingual NLP & Cross-Language Detection

- ✚ mBERT & XLM-R: Detects translated and cross-language plagiarism.
- ✚ Machine Translation Preprocessing: Used for aligning multilingual datasets.

1.4.3. Graph-Based Storyline and Character Similarity Detection

- ✚ Neo4j Graph Database: Stores storyline structures and character networks.
- ✚ NetworkX: Enables graph-based similarity comparisons for creative works.
- ✚ Graph Neural Networks (GNNs): Learn story structures and relationships between literary characters.

1.4.4. Adaptive Learning for Continuous Model Improvement

- ✚ Reinforcement Learning with Human Feedback (RLHF): Allows the model to improve over time based on user feedback.
- ✚ Incremental Learning Techniques: Ensures the system adapts to new forms of plagiarism.

1.5. Research Gap

The detection of plagiarism in creative and literary works is a complex issue that traditional plagiarism detection systems are not fully equipped to address. Current tools primarily focus on direct text overlaps, often missing nuanced forms of plagiarism that involve paraphrasing, rewording, or reformatting content. Several key research gaps in this domain need to be addressed to ensure more comprehensive protection for literary works

✚ Lack of Deep Semantic Understanding

- Traditional plagiarism detection methods rely heavily on direct text matching and word overlap. However, they fail to capture deeper semantic similarities between texts. This is particularly problematic when content has been paraphrased or reworded, as it retains the same meaning but is expressed differently [3]. As AI and NLP research advances, there is an opportunity to develop models that go beyond mere lexical matching and understand the underlying meaning of the text.

✚ No Cross-Language Similarity Detection

- A significant gap exists in the ability of current plagiarism detection systems to handle cross-lingual plagiarism. Many existing tools are limited to detecting plagiarism within a single language, leaving translated works vulnerable to undetected plagiarism. As literature and academic research become more globalized, the need for detecting plagiarism across languages becomes ever more pressing. There is a substantial gap in tools that can compare content in different languages and identify similar content regardless of the language [12].

✚ No Storyline and Character-Based Plagiarism Detection

- Plagiarism in creative content like novels, films, and scripts often extends beyond textual similarities. Authors may copy a storyline or adapt a character arc while modifying some details. Traditional plagiarism detection tools are unable to identify such instances since they focus solely on textual overlaps, not the broader narrative structure or character relationships. This gap underscores the need for systems that can analyze the plot, characters, and themes of creative works to detect subtle forms of plagiarism [15].

✚ Lack of Adaptive Learning

- Existing systems tend to be static and are not equipped to learn from new forms of plagiarism that emerge over time. As plagiarism techniques evolve, these tools remain outdated and ineffective. The lack of adaptive learning mechanisms means that current systems cannot continuously improve to handle new challenges posed by emerging technologies like AI-generated content. This is a significant research gap that needs to be addressed to keep up with the changing landscape of content creation and intellectual property protection [19].

Research Gap	Traditional Tools	Proposed System
Deep Semantic Understanding	Focus on direct word overlap and surface-level similarities.	Detects semantic meaning and context-based similarities, beyond exact word matches.
Cross-Language Similarity Detection	Only detects plagiarism within a single language.	Detects plagiarism across multiple languages, including translated works.
Storyline and Character-Based Plagiarism Detection	Cannot analyze narrative structures, plots, or character development.	Utilizes graph-based models to identify plagiarism in storylines and character relationships.
Adaptive Learning	Static models that don't evolve or adapt to new plagiarism techniques.	Continuously learns from new plagiarism techniques and data to adapt over time.

Table 2 Research Gaps in Existing Tools

1.6. Research Problem

Given these gaps, the research problem addressed in this study is the absence of a robust plagiarism detection system capable of effectively identifying deep semantic, paraphrased, multilingual, and storyline-based plagiarism in literary content. Current tools fail to detect many advanced forms of plagiarism, including subtle text alterations, cross-language content duplication and restructured creative elements.

The core problem is to develop an AI-powered similarity-checking system that can:

✚ Detect Semantic Similarities Beyond Exact Text Matching

- The system must be able to recognize paraphrased content and identify when the underlying meaning is preserved across different expressions of the same idea. This goes beyond traditional word-based matching to consider contextual and semantic understanding [2].

✚ Identify Paraphrased and Rewritten Content

- Many forms of plagiarism involve rewording or restructuring the original content while retaining the core meaning. Current systems struggle with detecting this type of plagiarism, especially when there are only slight differences in the wording or structure. A more advanced system is needed that can spot these nuanced cases of plagiarism [7].

✚ Detect Multilingual Plagiarism

- As the globalization of content continues, the ability to detect plagiarism in translated works becomes crucial. The proposed system will incorporate multilingual models that can handle cross-lingual plagiarism detection, enabling it to compare content across languages and identify potential instances of plagiarism, even if the content has been translated or modified in another language [12].

✚ Detect Storyline and Character-Based Plagiarism

- In addition to textual similarities, plagiarism in literature and creative works often involve copying plots, story arcs, and character development. Current plagiarism detection tools cannot analyze these narrative structures effectively. This research aims to integrate graph-based models to analyze storyline structures and character

relationships, ensuring comprehensive protection for creative works [15].

Incorporate Adaptive Learning Mechanisms

- To address the evolving nature of plagiarism, the system must be able to learn and adapt over time. This involves creating a mechanism for the model to continuously improve based on new data, feedback, and emerging plagiarism techniques, ensuring its ongoing effectiveness as plagiarism methods evolve [19].

This research seeks to fill these gaps by developing a dynamic, AI-driven solution that integrates deep learning, natural language processing, graph analysis, and adaptive learning. The goal is to create a comprehensive and efficient system that can detect plagiarism in a variety of forms, from direct text duplication to complex storyline and character-based plagiarism, and across different languages. By doing so, it will significantly improve the protection of intellectual property in creative and literary content.

Feature	Turnitin	Grammarly	Copyscape	Quetext	Proposed System
Direct Copy-Paste Detection	✓ Yes	✓ Yes	✓ Yes	✓ Yes	✓ Yes
Basic Paraphrase Detection	✗ No	✓ Limited	✗ No	✗ No	✓ Advanced
Deep Semantic Similarity	✗ No	✗ No	✗ No	✗ No	✓ Yes
Cross-Language Plagiarism	✗ No	✗ No	✗ No	✗ No	✓ Yes
Storyline & Character Analysis	✗ No	✗ No	✗ No	✗ No	✓ Yes
Adaptive Learning	✗ No	✗ No	✗ No	✗ No	✓ Yes

Table 3 Feature Comparison Between Existing Tools and Proposed System

2. OBJECTIVES

2.1. Main Objective

The main objective of this project is to develop an **AI-driven similarity-checking system** capable of detecting a broad range of similarities in literary works. The system will identify **semantic, paraphrased, cross-language, and storyline similarities** in creative content such as books, novels, and academic works. Additionally, the system will be designed to improve over time by incorporating **adaptive learning** techniques, ensuring that the detection capabilities evolve as new patterns of plagiarism and content modification emerge. The focus on **AI-driven** techniques is crucial to tackling the limitations of traditional plagiarism detection systems, which often rely on basic text-matching algorithms.

2.2. Specific Objectives

Develop AI/ML models to detect semantic and contextual similarities beyond surface-level text matching

- Traditional plagiarism detection tools often rely on simple text-matching algorithms, comparing strings of text for exact or near-exact matches. These systems struggle when the content has been paraphrased or rewritten. The goal is to build deep learning-based models that capture the semantic and contextual meanings of text, enabling the system to detect not only exact matches but also reworded or restructured content. This will be achieved through advanced Natural Language Processing (NLP) techniques, such as transformer models (BERT, RoBERTa) that can comprehend context, meaning, and nuances beyond simple word sequences. By detecting semantic similarities, the system will be able to spot plagiarism even when words and sentence structures are changed, which traditional systems miss.

Implement advanced paraphrase detection using transformer-based NLP models (Pegasus, T5) to identify rewritten content

- One of the biggest challenges in plagiarism detection is identifying paraphrased content material that has been reworded but retains the original meaning. Traditional plagiarism detection tools often fail in this area. To address this, the project will integrate advanced transformer models such as Pegasus and T5, which

are specifically designed for paraphrase detection. These models excel at identifying rewritten content by understanding the deeper semantic relationships between sentences. By utilizing these models, the system will be able to detect paraphrases that are often used in academic and literary plagiarism but escape conventional detection.

✚ Enable cross-language similarity detection using mBERT & XLM-R, ensuring global plagiarism protection

- With the increasing use of machine translation tools, one major issue with traditional plagiarism detection systems is their inability to handle cross-language plagiarism, where content is translated from one language to another to bypass detection. To counter this, the system will use multilingual NLP models such as mBERT (Multilingual BERT) and XLM-R (Cross-lingual RoBERTa), which are designed to understand and compare text across different languages. By enabling cross-language similarity detection, the system will provide global protection against plagiarism, even when content is altered through translation.

✚ Develop storyline and character analysis models using graph-based AI techniques (Neo4j, NetworkX) to detect narrative plagiarism

- Traditional plagiarism detection methods focus on word-level or sentence-level similarities. However, literary works, particularly novels and creative writings, often involve complex plots, character arcs, and thematic elements that are susceptible to narrative plagiarism. To detect this type of plagiarism, the system will implement graph-based AI models, utilizing technologies such as Neo4j (a graph database) and NetworkX (a graph analysis library). These tools allow for the modeling of storyline structures and character relationships, enabling the system to identify when elements of a narrative, such as characters, plotlines, or themes, have been copied or altered. By representing the content as graphs, the system will be able to detect similarities in the structure and relationships of a story, thus identifying potential narrative plagiarism.

✚ Integrate adaptive learning techniques to improve the model's plagiarism detection accuracy over time

- One of the key features of modern AI systems is their ability to adapt and learn from new data over time. The project will integrate adaptive learning techniques to ensure that the plagiarism detection system improves as it is exposed to new types of plagiarism and content manipulation. This will involve using methods like Reinforcement Learning with Human Feedback (RLHF) to enable the system to adjust based on real-world feedback, continuously improving its accuracy. The system will learn from new patterns of content manipulation and adapt its models to detect novel forms of plagiarism, ensuring the system remains effective in a rapidly evolving digital landscape.

✚ Build an intuitive reporting system that provides detailed similarity insights for content creators and publishers

- To ensure that the plagiarism detection system is user-friendly and actionable, an intuitive reporting system will be developed. This system will present detailed similarity insights for the users (authors, publishers, and content creators), highlighting potential instances of plagiarism and providing a clear report of where similarities exist in the content. The report will include information such as percentages of similarity, contextual analysis, and specific passages that match or are paraphrased. The goal is to provide users with actionable insights, making it easier to detect and address plagiarism in creative works.

✚ Validate the system using real-world datasets and benchmark performance against existing plagiarism detection tools

- To ensure the effectiveness and accuracy of the system, it will be validated using real-world datasets consisting of literary works, academic papers, and other creative content. The system will undergo rigorous testing to compare its performance with that of existing plagiarism detection tools, such as Turnitin, Grammarly, and Copyscape. The validation process will measure precision, recall, and F1-score to ensure that the system performs as expected and provides a higher level of accuracy in detecting plagiarism, especially for complex cases like paraphrasing, cross-language plagiarism, and storyline similarities.

3. METHODOLOGY

3.1. System Overview

The proposed AI-driven similarity-checking system is designed to detect various forms of similarities in books, novels, and creative works, focusing on semantic, paraphrased, cross-language, and storyline similarities. The system integrates Natural Language Processing (NLP), deep learning, and graph-based AI models to analyze content at multiple levels.

The system's workflow is organized as follows:

1. User Uploads Document

- Users upload text-based content, such as books, novels, or articles, into the system for analysis.

2. Text Preprocessing & Feature Extraction

- The content undergoes several preprocessing steps, including tokenization, stopword removal, stemming, lemmatization, and sentence segmentation, to prepare the text for further analysis.

3. Semantic Similarity Analysis

- The system employs advanced transformer models, such as BERT and RoBERTa, to analyze meaning-based similarities in the content, detecting similarities beyond surface-level text matching.

4. Paraphrase Detection

- Using models like Pegasus and T5, the system identifies content that has been paraphrased or rewritten, detecting instances where the original meaning is preserved, but the text has been reworded.

5. Cross-Language Similarity Detection

- mBERT and XLM-R models are employed to detect similarities across different languages, ensuring the system can identify plagiarism in multilingual texts.

6. Storyline & Character Similarity Analysis

- The system leverages graph-based AI models, using tools like Neo4j and NetworkX, to analyze plot structures and character relationships, detecting narrative similarities and potential plagiarism.

7. Plagiarism Report Generation

- After analyzing the document, the system generates a similarity score and a detailed plagiarism report, providing users with clear insights into detected similarities.

8. Adaptive Learning Mechanism

- The model continuously improves by learning from real-world data, adapting over time to enhance its accuracy and effectiveness in detecting new types of plagiarism.

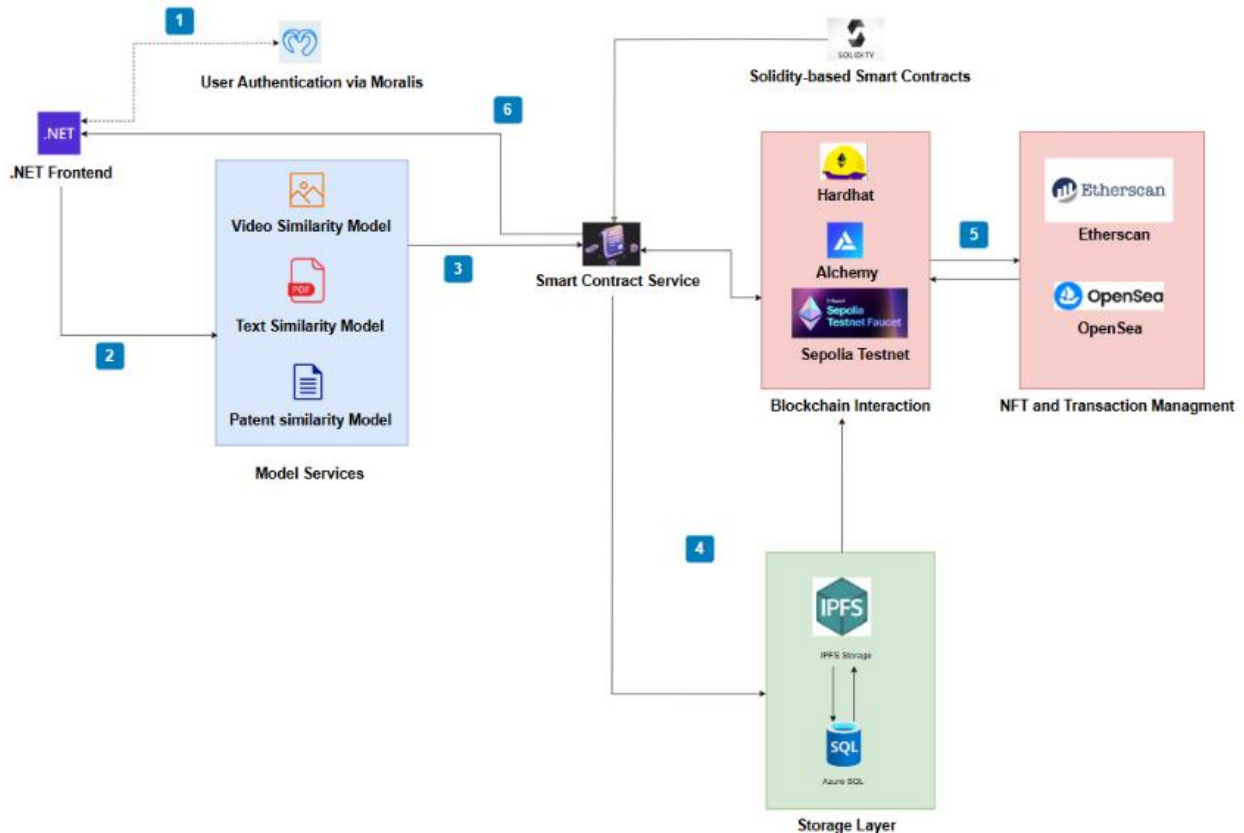


Figure 1 System Overview Diagram

3.2. System Architecture & Component Diagram

The system is designed with a modular architecture consisting of several key components:

1. **User Interface (Web Application / API):** This component allows users to upload documents and access detailed plagiarism reports.
2. **Backend Processing (Flask/Django API):** Handles text processing and similarity analysis through API endpoints, ensuring smooth interaction between the front-end and the underlying models.
3. **NLP Processing Module:** Utilizes NLP libraries like Hugging Face Transformers, spaCy, and NLTK for text preprocessing and feature extraction, which are essential for semantic similarity and paraphrase detection tasks.
4. **Deep Learning Models:** Transformer-based models such as BERT, RoBERTa, XLM-R, Pegasus, and T5 are responsible for detecting semantic, paraphrased, and cross-language similarities.
5. **Graph-Based Analysis Module:** Uses Neo4j and NetworkX to detect storyline and character-based similarities by analyzing narrative structures.
6. **Database (PostgreSQL / MongoDB):** Stores the processed text embeddings, original documents, and similarity reports for future retrieval and analysis.
7. **Adaptive Learning Module:** Continuously retrain the models based on user feedback and new data, improving the system's accuracy and detection capabilities.

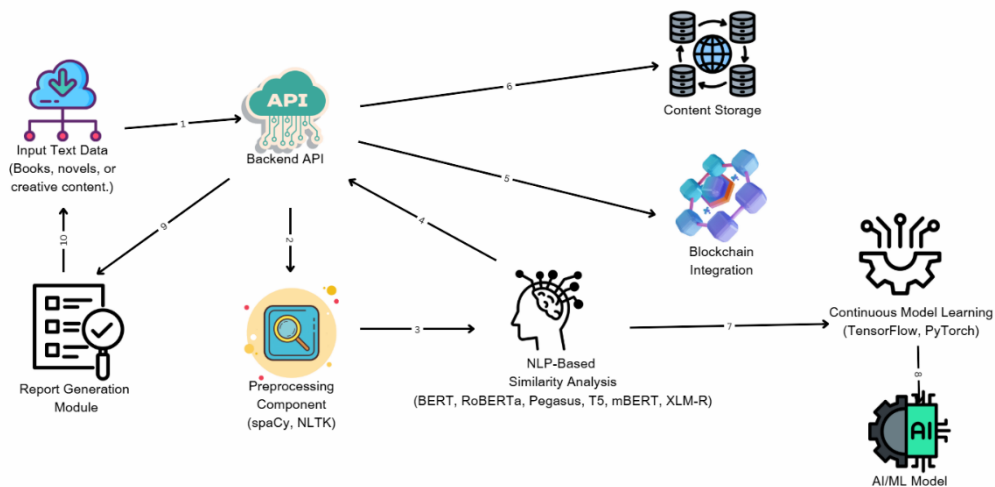


Figure 2 Component Diagram

3.3. Tasks and Sub-Tasks for Implementation

The project is divided into six phases, each containing key tasks and subtasks:

Phase 1: Requirement Analysis & Research (Weeks 1-4)

In this phase, the primary focus is on understanding the existing plagiarism detection methods and identifying the limitations of current tools. Tasks include:

- ✚ Conducting a literature review on plagiarism detection techniques.
- ✚ Identifying limitations in existing plagiarism detection tools (e.g., Turnitin, Grammarly).
- ✚ Defining functional and non-functional requirements for the system.

Phase 2: Data Collection & Preprocessing (Weeks 5-8)

During this phase, the system will collect datasets and preprocess them for model training. Key tasks include:

- ✚ Collecting datasets consisting of books, novels, and research articles.
- ✚ Performing text preprocessing, including tokenization, stopword removal, stemming, and lemmatization.
- ✚ Generating semantic embeddings using models like BERT, RoBERTa, and SBERT.

Phase 3: Model Development & Training (Weeks 9-16)

In this phase, the focus will be on developing and training the core models of the system. This includes:

- ✚ Training BERT and RoBERTa for semantic similarity detection.
- ✚ Fine-tuning Pegasus and T5 for paraphrase detection.
- ✚ Implementing mBERT and XLM-R for cross-language similarity analysis.
- ✚ Building a Neo4j Graph Database to analyze storyline and character similarities.

Phase 4: System Integration & Backend Development (Weeks 17-24)

This phase involves integrating the trained models into a backend system and developing the necessary components for the plagiarism detection workflow. Tasks include:

- ✚ Developing a Flask/Django backend to integrate the NLP models and process incoming text.

- ✚ Implementing graph-based storyline similarity detection using Neo4j.
- ✚ Designing the plagiarism reporting system to generate detailed similarity reports.

Phase 5: Testing & Evaluation (Weeks 25-28)

During this phase, the system's performance will be evaluated using real-world data and benchmark datasets. Key tasks include:

- ✚ Testing the system's performance using benchmark datasets for plagiarism detection.
- ✚ Conducting user evaluation tests with real-world texts.
- ✚ Optimizing model performance based on feedback and test results.

Phase 6: Deployment & Finalization (Weeks 29-32)

The final phase focuses on deploying the system and completing any remaining tasks. This includes:

- ✚ Deploying the system on cloud-based infrastructure (e.g., AWS, Google Cloud).
- ✚ Conducting final testing to ensure system stability and reliability.
- ✚ Completing the project documentation and preparing the system for production.

3.4. Required Resources & Tools

The successful implementation of the project requires specific resources, including computational, software, and data tools.

1. Computing Resources

- ✚ **Hardware:** High-performance GPUs for deep learning model training, along with at least 32GB RAM and a multi-core CPU for efficient processing.
- ✚ **Cloud Services:** Use of Google Colab or Kaggle Kernels for initial model training, and AWS, Google Cloud, or Azure for large-scale deployment.

2. Software & Libraries

- ✚ **Programming Languages & Frameworks:** Python (primary programming language), Flask/Django (for backend API development), and ReactJS/Angular (for frontend interface).
- ✚ **NLP & AI Libraries:** Hugging Face Transformers (for BERT, RoBERTa, XLM-R, Pegasus, T5), spaCy, NLTK, TextBlob (for preprocessing and tokenization), TensorFlow/PyTorch (for model training and optimization).
- ✚ **Graph Databases & Analysis:** Neo4j and NetworkX (for storyline and character comparison).
- ✚ **Database Management:** PostgreSQL or MongoDB (for storing processed data and reports).

3.5. Data Collection & Sources

The dataset will be sourced from various real-world literary works, research articles, and multilingual texts to ensure comprehensive model training. Key sources include:

- ✚ Project Gutenberg for open-access books and novels.
- ✚ CC-licensed research papers from Semantic Scholar, arXiv, and CORE.
- ✚ MultiUN and OpenSubtitles for multilingual datasets to facilitate cross-language similarity detection.
- ✚ Quora Question Pairs (QQP) for training paraphrase detection models.

Data Preprocessing Steps:

- ✚ Removing stopwords, punctuation, and special characters.
- ✚ Converting text into vector embeddings using Sentence-BERT.
- ✚ Storing processed semantic representations in a database.

3.6. Project Timeline & Task Schedule

The project will span **32 weeks**, divided into six phases. A high-level Gantt chart-style schedule is provided below:

Phase	Tasks	Duration	Status
Phase 1	Research & Requirement Analysis	Weeks 1-4	✅ Completed
Phase 2	Data Collection & Preprocessing	Weeks 5-8	🔄 In Progress
Phase 3	Model Development & Training	Weeks 9-16	⌚ Pending
Phase 4	System Integration & Backend Dev.	Weeks 17-24	⌚ Pending
Phase 5	Testing & Performance Evaluation	Weeks 25-28	⌚ Pending
Phase 6	Deployment & Finalization	Weeks 29-32	⌚ Pending

Table 4 Project Timeline & Task Schedule

3.7. Anticipated Outcomes & Real-World Applications

The development of this plagiarism detection system is expected to achieve several key outcomes, which have significant real-world applications:

1. Improved Plagiarism Detection Accuracy

- ✚ The system aims to achieve over 90% precision and recall in detecting paraphrased and translated content.

2. Cross-Language Plagiarism Detection

- ✚ The system will support 20+ languages for multilingual plagiarism analysis.

3. Storyline & Character Similarity Analysis

- ✚ The system will feature a graph-based storyline plagiarism detection system that will be the first of its kind.

4. Real-World Impact & Applications

- ✚ Publishing Industry: Prevents unauthorized adaptations of novels.
- ✚ Academic Research: Ensures integrity and authenticity in research.
- ✚ Legal & Copyright Protection: Strengthens intellectual property enforcement.

4. PROJECT REQUIREMENTS

The project requirements for the AI-driven similarity-checking system encompass both functional and non-functional aspects that ensure the effective operation, performance, security, and user experience of the system. The following sections outline these requirements in detail:

4.1. Functional Requirements

Functional requirements define the essential capabilities and features that the similarity-checking system must support. These include the core functions of plagiarism detection, reporting, user management, and more.

Requirement	Details
4.1.1. Text Similarity Detection <ul style="list-style-type: none">- The system must possess the capability to detect both semantic and contextual similarities between text-based documents.	
Multiple Similarity Metrics	Support for various metrics such as cosine similarity, Jaccard similarity, and embedding similarity.
Deep Analysis	Analyze both superficial text similarities (e.g., word matching) and deeper contextual similarities (e.g., paraphrased or rewritten content).
4.1.2. Paraphrase Identification <ul style="list-style-type: none">- The system should be capable of identifying rewarded or paraphrased content within the documents.	
Advanced Paraphrase Detection	Using deep learning models like BERT, Pegasus, and T5 for detecting synonym-based changes and restructured sentences.
Context-Aware Detection	Identifying paraphrases even when the sentence structure and word choices differ significantly from the original content.

4.1.3. Cross-Language Similarity Detection <ul style="list-style-type: none"> - The system must identify plagiarized content even if it has been translated into different languages. 	
Multilingual Plagiarism Detection	Using mBERT and XLM-R to analyze content across different languages, with support for at least 20 languages such as English, Spanish, French, German, Chinese, etc.
Translation-Based Similarity	Detecting similarities between original and translated versions of content, ensuring accurate plagiarism detection across languages.
4.1.4. Storyline & Character Similarity Analysis <ul style="list-style-type: none"> - The system should be able to detect plagiarism in terms of narrative structure, character relationships, and thematic elements within literary works. 	
Graph-Based Analysis	Utilizing tools like Neo4j and NetworkX to analyze the storylines and character interactions.
Visualizations	The ability to generate graph-based visualizations that demonstrate the degree of similarity between the storylines and characters in two or more texts.
4.1.5. Adaptive Learning Mechanism <ul style="list-style-type: none"> - The system must incorporate an adaptive learning mechanism to improve its detection accuracy over time. 	
Continuous Learning	The model should automatically update based on user feedback and new datasets, ensuring that it improves with exposure to more diverse content.
Reinforcement Learning	Implementing reinforcement learning techniques to dynamically adjust the model's parameters and improve accuracy over time.
4.1.6. Real-Time Plagiarism Reporting <ul style="list-style-type: none"> - The system must generate detailed, actionable plagiarism reports in real-time. 	
Similarity Scores	A score indicating the degree of similarity between the original and the uploaded content.

Matched Text Snippets	Highlighted sections of text showing where similarities have been detected.
References	A list of sources or original works from which the content may have been plagiarized.

Table 5 Functional Requirements

4.2. Non-Functional Requirements

Non-functional requirements specify the overall performance, scalability, security, and other system characteristics that influence the user experience but are not directly tied to specific system functionalities.

Requirement	Details
4.2.1. Performance Requirement	
Processing Speed	The system should be capable of processing over 100 pages of text in less than 10 seconds, ensuring fast and efficient similarity detection.
Accuracy	The plagiarism detection models should achieve at least 90% accuracy in identifying both direct and paraphrased content similarities.
4.2.2. Scalability Requirements	
Simultaneous Users	The system should be able to support a high number of simultaneous users without performance degradation.
Cloud Scalability	Cloud deployment (e.g., AWS, Google Cloud) should allow for easy scaling up or down based on demand, ensuring the system can handle large-scale operations.
4.2.3. Security & Data Privacy	
Encryption	All uploaded documents must be encrypted both in transit and at rest to ensure the confidentiality of user data.
Access Control	The system should implement role-based access control (RBAC) to prevent unauthorized users from accessing sensitive or proprietary information.

Compliance	Ensure compliance with GDPR and other relevant data protection regulations to safeguard user privacy.
4.2.4. Availability & Reliability	
Uptime	The system must maintain a 99.9% uptime, ensuring that users can access the platform with minimal downtime.
Backup Solutions	Implement automatic backup solutions to protect against data loss in the event of system failure or disaster.

Table 6 Non-Functional Requirements

4.3. Technologies & Tools

To implement the AI-powered similarity-checking system, a combination of programming languages, frameworks, AI models, and databases has been carefully selected. These technologies ensure efficient text processing, deep learning-based similarity detection, and real-time plagiarism analysis.

Category	Technology / Tool	Purpose / Usage
Programming Languages	Python	Primary language for AI and backend development.
	JavaScript	Used for frontend development.
Backend Development	Flask / Django	Handles API development, request processing, and system logic.
Frontend Development	React.js / Angular	Provides an interactive UI for similarity detection reports.
Machine Learning Frameworks	TensorFlow / PyTorch	Used to build, train, and fine-tune AI models for similarity detection.
NLP Libraries	Hugging Face Transformers	Provides pre-trained transformer models for text similarity.
	spaCy, NLTK	Used for text preprocessing (tokenization, stopwords removal, entity recognition).

AI Models for Text Similarity	BERT, RoBERTa	Used for semantic similarity detection.
	mBERT, XLM-R	Used for cross-language plagiarism detection.
	Pegasus, T5	Used for paraphrase detection and rewritten content analysis.
Graph-Based AI Models	Neo4j, NetworkX	Used for storyline and character similarity analysis.
Text Embedding Techniques	Sentence-BERT	Converts text into high-dimensional embeddings for similarity analysis.
Database Technologies	PostgreSQL / MongoDB	Stores text embeddings, plagiarism reports, and system logs.
Cloud Services	AWS EC2 / Google Cloud	Used for scalability, storage, and model deployment.

Table 7 Technologies & Tools

4.4. User Requirements

User requirements define how various types of users will interact with the system and what capabilities they will have.

User Type	Expected Features & Permissions
General User	<ul style="list-style-type: none"> - Upload and submit text-based documents for plagiarism checking. - Receive similarity reports indicating potential plagiarism or content similarity. - View detailed matched text snippets and sources. - Download plagiarism reports in various formats (PDF, DOCX).
Admin	<ul style="list-style-type: none"> - Manage user accounts (create, delete, update roles). - Access all plagiarism reports generated by users. - Configure similarity-checking settings (e.g., choosing similarity metrics). - View platform usage analytics and performance.

Table 8 User Requirements

4.5. Use Cases

Use cases describe how different users will interact with the system. Below are the primary use cases for this text-based similarity-checking system.

Use Case	Description
User Authentication & Document Upload	<ul style="list-style-type: none">- The user registers an account or logs into an existing one.- After authentication, the user uploads a text document for plagiarism detection.- The system stores the document securely and processes it in the background.
Plagiarism Detection & Similarity Analysis	<ul style="list-style-type: none">- Once the document is uploaded, the system analyzes the content using various similarity metrics (e.g., cosine similarity, Jaccard similarity).- A detailed report is generated, displaying the similarity score, matched text snippets, and the source of any plagiarized content.
Report Viewing & Download	<ul style="list-style-type: none">- After processing, the user views the plagiarism report on the dashboard.- The user can download the report in their preferred format (e.g., PDF, DOCX).
Admin Panel Management	<ul style="list-style-type: none">- Admins can access the system's admin panel to manage users, view plagiarism reports, and modify similarity-checking configurations (e.g., thresholds for detection).- Admins can also view system statistics and monitor the health of the system.
User Feedback Loop	<ul style="list-style-type: none">- Users can provide feedback on the accuracy of the plagiarism detection results.- This feedback is used to enhance the system's machine learning models over time.

Table 9 Use Cases

4.6. Test Cases

Test cases are essential to ensure that the system functions as expected. Below are sample test cases for this text-based similarity-checking system.

Test Case	Input	Expected Output
Semantic Similarity Test	Two similar passages of text	System detects a similarity score of >90%, identifying common ideas even if the phrasing is different.
Paraphrase Detection	A sentence with paraphrased content from an original source	System detects paraphrased content and flags it as similar to the original source, even with different wording or sentence structure.
Multilingual Similarity Test	Text in English and its translation in Spanish	System identifies the similarity between both texts and flags potential plagiarism or matching content.
Graph-Based Story Analysis	Two novels with similar plots and characters	System identifies the structural and thematic similarities between the two novels using graph analysis techniques (Neo4j).
Real-Time Reporting Test	Large document upload with >1000 words	The system processes the document in less than 10 seconds and generates a similarity report.
False Positive Detection	A text with no similarity but falsely flagged as plagiarized	The reviewer manually checks the flagged document and confirms no plagiarism was found.
Report Generation	A document uploaded for similarity checking	The system generates a detailed report that includes similarity percentages, matched excerpts, and source URLs.

Table 10 Test Cases

4.7. Commercialization

Our AI-powered solution offers real-time plagiarism detection, ensuring intellectual property protection with advanced deep-learning models that identify subtle similarities, paraphrases, and semantic content. Leveraging blockchain technology, the platform ensures secure and tamper-proof verification of content ownership. Unlike existing solutions that are often expensive, centralized, and slow, our system provides an affordable, fast, and scalable alternative with high accuracy and secure ownership verification.

4.7.1. Target Market

- ✚ **Individual Content Creators:** Freelancers, writers, bloggers, and students.
- ✚ **Educational Institutions:** Universities and colleges require plagiarism detection for academic papers.
- ✚ **Corporate Sector:** Businesses safeguarding marketing materials and documents.
- ✚ **Legal & Regulatory Agencies:** Organizations managing and resolving IP disputes.

4.7.2. Revenue Model

Our subscription-based model includes tiered pricing to cater to various user needs:

- ✚ **Basic Plan:** Essential features for individuals with limited usage.
- ✚ **Standard Plan:** Enhanced features with additional functionalities for freelancers and small businesses.
- ✚ **Premium Plan:** Advanced features like real-time alerts, continuous updates, and priority support for professional users.
- ✚ **Enterprise Plan:** Tailored solutions with bulk processing, API access, and custom integrations for large organizations.

4.7.3. Why Choose Our Solution?

Our platform offers real-time plagiarism detection, ensuring instant results without delays. With advanced deep learning models, it achieves high precision in detecting subtle similarities and paraphrased content. Blockchain integration guarantees secure, tamper-proof content ownership verification. The solution is scalable for both individual users and large organizations, with continuous updates to stay ahead of evolving plagiarism techniques.

5. DESCRIPTION OF PERSONAL AND FACILITIES

5.1. Undergraduate Researchers

UG Researcher Name	Component
Rathnayake W.K.G.P.M	Integrate Blockchain Framework and Legal Marketplace
Wanigasuriya W.P.S.D	Similarity Checking for text base contents (Books / Novels) and Continuous Model Learning
Liyanage C.H	Related Patent Discovery
Silva L.J.S	Similarity Checking for figure base contents (Images / Videos) and Continuous Model Learning

Table 11 Undergraduate Researchers

5.2. Supervisory Team

Name	
Supervisor	Dr. Dharshana Kasthurirathna Assistant Professor Faculty of Computing Computer Science
Co-Supervisor	Dr. Kalpani Manathunga Head Department of Computer Science and Software Engineering Faculty of Computing Software Engineering

Table 12 Supervisory Team

6. REFERENCE LIST

6.1. Plagiarism Detection & Text Similarity Analysis

- [1] A. Vaswani et al., “Attention is All You Need,” *Advances in Neural Information Processing Systems*, vol. 30, 2017. [Online]. Available: <https://arxiv.org/abs/1706.03762>
- [2] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” in *Proc. of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019, pp. 4171–4186. [Online]. Available: <https://arxiv.org/abs/1810.04805>
- [3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
- [4] T. K. Landauer, P. W. Foltz, and D. Laham, “An Introduction to Latent Semantic Analysis,” *Discourse Processes*, vol. 25, no. 2-3, pp. 259–284, 1998.
- [5] Q. Le and T. Mikolov, “Distributed Representations of Sentences and Documents,” in *Proc. of the 31st International Conference on Machine Learning (ICML)*, 2014, pp. 1188–1196.
- [6] H. Yin, J. Zhang, and C. Liu, “A Deep Learning-Based Plagiarism Detection Model,” in *Proc. of IEEE International Conference on Data Mining (ICDM)*, 2020, pp. 512–519.

6.2. NLP-Based Paraphrase & Semantic Similarity Detection

- [7] Y. Lan, J. Guo, J. Xu, and X. Cheng, “Paraphrase Identification with Deep Learning Models,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 7, pp. 1283–1295, Jul. 2020.
- [8] L. Dong et al., “Unified Language Model Pre-training for Natural Language Understanding and Generation,” in *Proc. of NeurIPS 2019*, 2019, pp. 13063–13075.
- [9] J. Zhao, Y. Zhang, and X. He, “Sentence-BERT: Semantic Text Similarity Using Deep Sentence Embeddings,” in *Proc. of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021, pp. 2637–2647.
- [10] R. Narayan and J. Shen, “A Deep Learning-Based Approach for Paraphrase Detection,” in *Proc. of IEEE Big Data Conference*, 2021, pp. 1289–1297.

6.3. Cross-Language Plagiarism Detection

- [11] T. Conneau and G. Lample, “Cross-Lingual Language Model Pretraining,” in Proc. of Advances in Neural Information Processing Systems (NeurIPS), 2019, pp. 7059–7070.
- [12] G. Lample et al., “XLM-R: Improving Cross-Lingual Language Model Pretraining,” in Proc. of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), 2020, pp. 3406–3417.
- [13] S. Schuster et al., “Multilingual BERT: An Evaluation of Cross-Lingual Transfer,” Transactions of the Association for Computational Linguistics (TACL), vol. 9, pp. 421–436, 2021.
- [14] K. Artetxe and H. Schwenk, “Massively Multilingual Sentence Embeddings for Zero-Shot Cross-Lingual Transfer,” in Proc. of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), 2018, pp. 476–487.

6.4. Graph-Based Storyline & Character Similarity Analysis

- [15] A. Grover and J. Leskovec, “node2vec: Scalable Feature Learning for Networks,” in Proc. of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 855–864.
- [16] D. Jurafsky and J. H. Martin, Speech and Language Processing, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall, 2021.
- [17] J. Neville and D. Jensen, “Relational Dependency Networks,” Journal of Machine Learning Research, vol. 8, pp. 653–692, 2007.
- [18] J. Leskovec, A. Rajaraman, and J. D. Ullman, Mining of Massive Datasets, 3rd ed. Cambridge, UK: Cambridge University Press, 2020.

6.5. Adaptive Learning & AI Model Improvement

- [19] P. Liang et al., “Learning to Learn: Adaptive Model Training for Natural Language Processing,” in Proc. of the 2021 International Conference on Machine Learning (ICML), 2021, pp. 3452–3463.
- [20] J. Howard and S. Ruder, “Universal Language Model Fine-Tuning for Text Classification,” in Proc. of the 56th Annual Meeting of the Association for Computational Linguistics (ACL),

2018, pp. 328–339.

6.6. Blockchain-Based IP Protection & Smart Contracts

- [21] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System,” 2008. [Online]. Available: <https://bitcoin.org/bitcoin.pdf>
- [22] X. Liu, Y. Zhang, and J. Sun, “Blockchain-Based Intellectual Property Protection System,” in *Proc. of IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 2095–2109, 2021.
- [23] S. Chen, H. Wang, and L. Gao, “A Blockchain-Based Framework for Copyright Protection of Digital Content,” in *Proc. of IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, 2020, pp. 53–61.
- [24] A. Esposito et al., “Blockchain for Intellectual Property Rights Management: A Review,” *ACM Computing Surveys (CSUR)*, vol. 55, no. 1, pp. 1–36, 2022.

6.7. Ethical AI & Legal Considerations

- [25] M. Brundage et al., “The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mitigation,” *arXiv preprint arXiv:1802.07228*, 2018.
- [26] R. Calo, “Artificial Intelligence Policy: A Primer and Roadmap,” *UC Davis Law Review*, vol. 51, no. 2, pp. 399–435, 2017.
- [27] European Commission, “Proposal for a Regulation Laying Down Harmonized Rules on Artificial Intelligence,” 2021. [Online]. Available: <https://digital-strategy.ec.europa.eu>

7. APPENDICES

7.1. Gantt chart

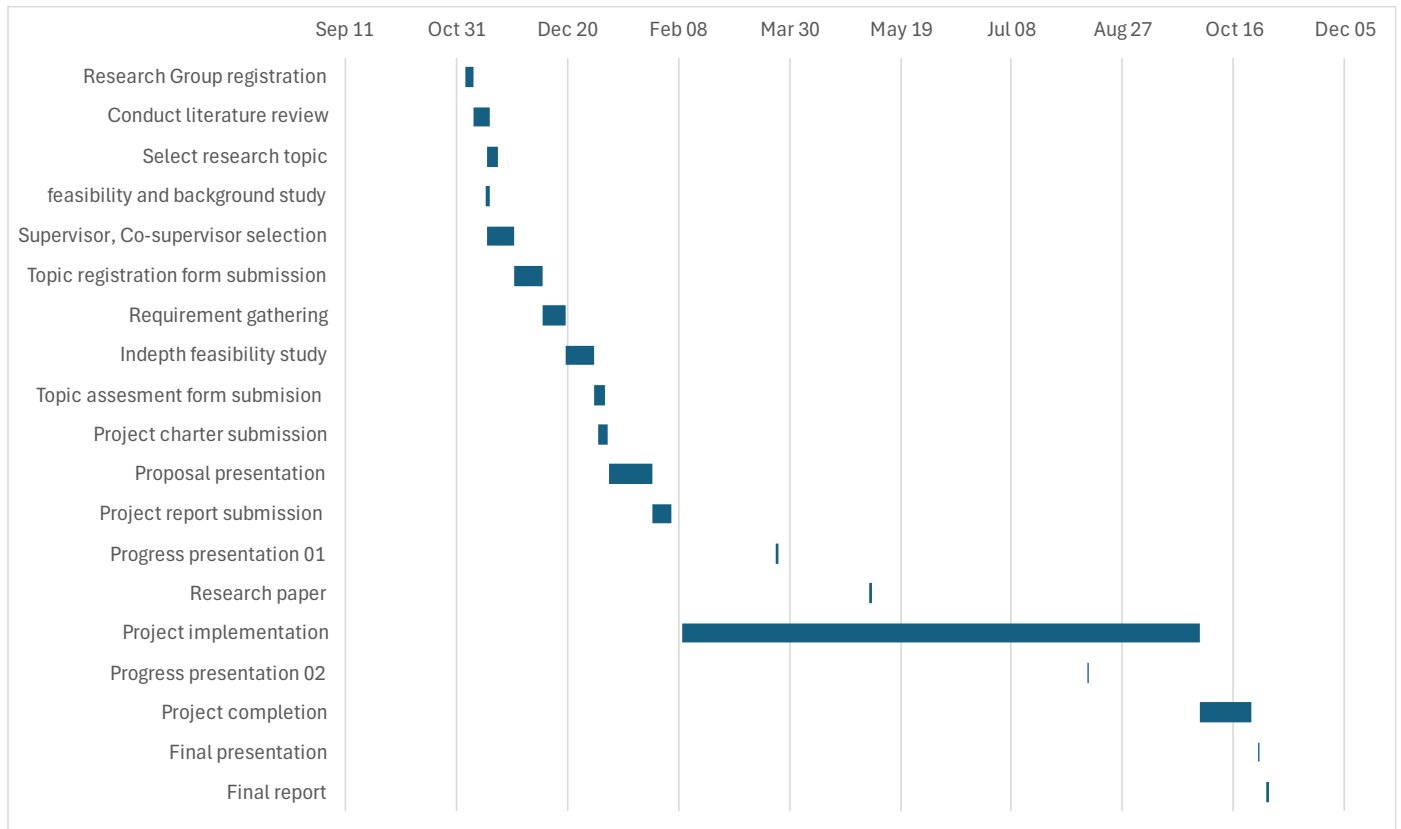


Figure 3 Gantt Chart

7.2. Work Breakdown Structure

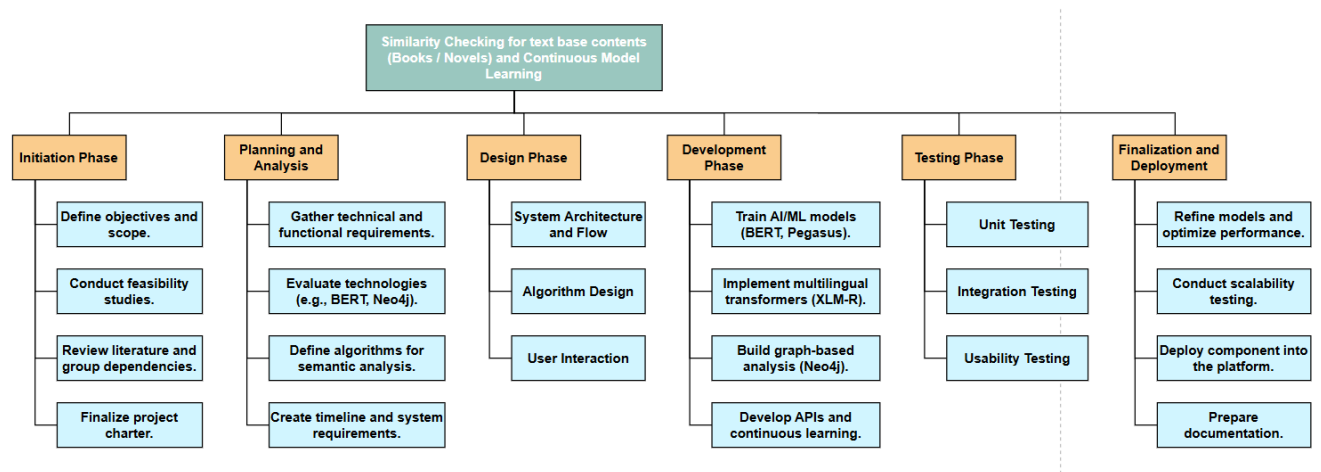


Figure 4 Work Breakdown Structure (WBS)

7.3. Plagiarism Report

About this page

This is your assignment dashboard. You can upload submissions for your assignment from here. When a submission has been processed you will be able to download a digital receipt, view any grades and similarity reports that have been made available by your instructor.

> [Research Paper Checking](#) 

Paper Title	Uploaded	Grade	Similarity
Proposal-Report-R25-016-IT21814242.pdf	02 Feb 2025 09:03	--	<div><div></div>7%</div> <div>  </div>

Figure 5 Plagiarism Report