International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 07 Issue: 04 | April - 2023 Impact Factor: 8.176 ISSN: 2582-3930

THE PLAGIARISM CHECKER USING MACHINE LEARNING

A.B.T.S. Bhavya, Harsha Vardhan S, Ch. Sai Sampath, Chetan Sai Abhishek I

Dr. P. Anuradha(Associate Professor)
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
BACHELOR OF TECHNOLOGY

GITAM
(Deemed to be University)
VISAKHAPATNAM

ABSTRACT

Plagiarism is one of the most increasing problems in many fields; the academic field is one of them. It comes in various forms, from replacing a word with its synonym to sentence modification, transformation, and many more. Though humans are known for their efficient working ability, they may not be able to detect plagiarism in all scenarios accurately. They can't see similarities against more than one million online documents in seconds. So, concluding that plagiarism detection is tedious and time-consuming work for a human, it would be nice to have a plagiarism checker to do plagiarism detection for us. In this project, a plagiarism checker application is developed based on machine learning as its core that searches a vast database for plagiarized content. Using the vector embeddings concept, a plagiarism checker application is developed, which converts data, for example, text data, into a list of numbers, thus allowing various operations to be performed on the converted data. Vectors are helpful because when we present real-world entities like audio, images, text, etc., as vector embeddings, the semantic similarity between these entities can be quantified by how close they're to each other as points in vectors. Models are trained to translate entities into vectors; NLP is commonly used for such training. These

vector embeddings will be added to the pre-processed database, which will then be ready to be used for our similarity check. The machine learning-based application will take text as input from the user, check the text against the database, and return all the articles from which the input text could be plagiarized, along with the match score.

Keywords: Plagiarism detection, machine learning, vector embeddings, NLP.

1. INTRODUCTION

Plagiarism is "taking someone's work/ideas and passing them off as their own without giving the deserved credit to the content owner." Plagiarism can be done in many forms, such as copying and pasting the content without giving credit to the original author, paraphrasing, translation plagiarism, i.e., translating content from one language to another, etc. Plagiarism has found its way into many areas, such as assignments, images, articles, newspapers, etc. With the growing use of the internet worldwide, having their students maintain integrity in their academic submissions has become one of the most significant challenges in all universities, schools, and educational centers. Plagiarism negatively affects a student's academic career. It leads to an unfair evaluation of assignments. Since manually checking for plagiarism in all the works is tedious, most universities have resorted to using plagiarism tools like Turnitin, Grammarly, etc., to let the machine scan the assignments for plagiarism. But, the accuracy of these tools mainly depends on the ability of the algorithm to detect the different types of plagiarism that an individual could do.

As part of letting the machine detect plagiarism, many models have been proposed for plagiarism detection, such as String matching, bag of word analyzing, Word2Vec, etc. In this context, we propose a Plagiarism Detection model that is based on the Embedding technique for vector embeddings and the Average Word Embeddings Model. Our model can detect various types of Plagiarism acts, including patch-written content, i.e., when an individual tries to mask the fact that they have copied the content by editing the original content. Our model is found to produce promising results.

2. LITERATURE REVIEW

Hiten Chavan et al. [1] discussed a plagiarism model that is free and user-friendly, particularly for instructors and educational institutions. Many built-in machine-learning technologies have been utilized to create this project. The author outlines the philosophy behind using the Sci-Kit package, which has several effective and practical machine-learning techniques. Their proposed approach for feature extraction from the text uses this library. Word embedding, or the transformation of textual data into an array of integers, was done using the Tf-idf vectorizer.

El Mostafa Hambi, Faouzia Benabbou, et al. [2] proposed a model comprising three layers: a preprocessing layer with word embedding, learning layers, and a detection layer. The author's proposed model was based on three models: Siamese Long Short-term Memory (SLSTM), Convolutional Neural Network (CNN), and Doc2vec for the research. Because of this, they successfully improved the job in a way that recommended an accuracy score of 98.33%. As a result, the evidence from the internet search was supported, and plagiarism was found to determine the similarity score.

Nishesh Awale, Mitesh Pandey, Anish Dulal, Bibek Timsina, et al. [3] suggested using machine learning to identify programming assignment plagiarism. Features pertaining to source code were computed using the n-gram similarity score, coding style similarity, and dead codes. The authors trained the datasets for their proposed model using the xgboost model, which also predicts if a pair of source codes is copied or not. This authors' study considers several unused variables and functions in the source code, which are typically disregarded in many plagiarism detection methods. On the test set, the authors' model had an average of 0.905(f1-score) and an accuracy score of 94%. As a result, the authors successfully compared the performance of the xgboost to that of the Support Vector Machine (SVM), and they showed that the xgboost model outperformed the SVM on the dataset.

Babitha V, Harshitha M, Hindumathi A, Reshma Farhin J, et al. [4] discuss a model that primarily focuses on detecting plagiarism in assignments. The authors' focused on the importance of the convenience of professors, while assessing the degree of plagiarism in students' assignments therefore, they proposed an approach that avoids the tiresome effort and increases speed and efficiency over the old manual method. The authors' model takes a file for inpu,t and this file goes through the process of Tokenization, Cleaning, Stop Word, and Stemming stages

during the preparation. Creating the report and analyzing plagiarism will follow.

3. PROBLEM IDENTIFICATION & OBJECTIVE

3.1 PROBLEM IDENTIFICATION

We can see that all the proposed Plagiarism detection models, while working efficiently, focus more on the model's performance. Still, it is essential to note that these models are complex and will probably require high computations, so in our proposed model, while focusing on producing maximum accuracy, we will also make the model as simple as possible and make the model always available for plagiarism detection to the user avoiding hassle like having to create an account unlike many plagiarism tools like Turnitin.

3.2 OBJECTIVE

The proposed model goal is to suggest a uniform method for using NLP to identify plagiarism. The traditional Word2Vec model, the GLoVe embedding model, and the Average Word Embedding Model are different ways of vector embedding. This model is trained with the help of the Average Word Embedding (AWE) Model.

The model proposes a novel approach to detect plagiarism that combines machine learning and natural language processing techniques. The three significant steps of this system's operation are text input, text conversion, similarity scoring, and report production. The model's primary goal is detecting plagiarism in digital documents. The study suggests a plagiarism detector unaffected by rearranging the problem statement or identifier order. It contrasts the viewpoint with that of a computer-generated plagiarism detector. The study employs the NLP-based vector embedding technique.

4. SYSTEM METHODOLOGY

4.1. UML DIAGRAMS

Unified Modeling Language, popularly known as UML, is a modeling language used by developers to visualize the design/behavior of an existing model or to describe the design/behavior of a model that has to be developed.

4.1.1 USE CASE DIAGRAM

A use case Diagram is a Behavioral Diagram whose primary purpose is to identify and visualize the actors and their role in the system. In our proposed model, we have two actors: the user and the machine. The user will be the one who will give the text that is to be tested for plagiarism to the machine; the machine takes the text and converts it into vectors to perform a similarity check against the data present in the vectors database and returns the similarity percentage and the articles link it is plagiarized from for the user to view.

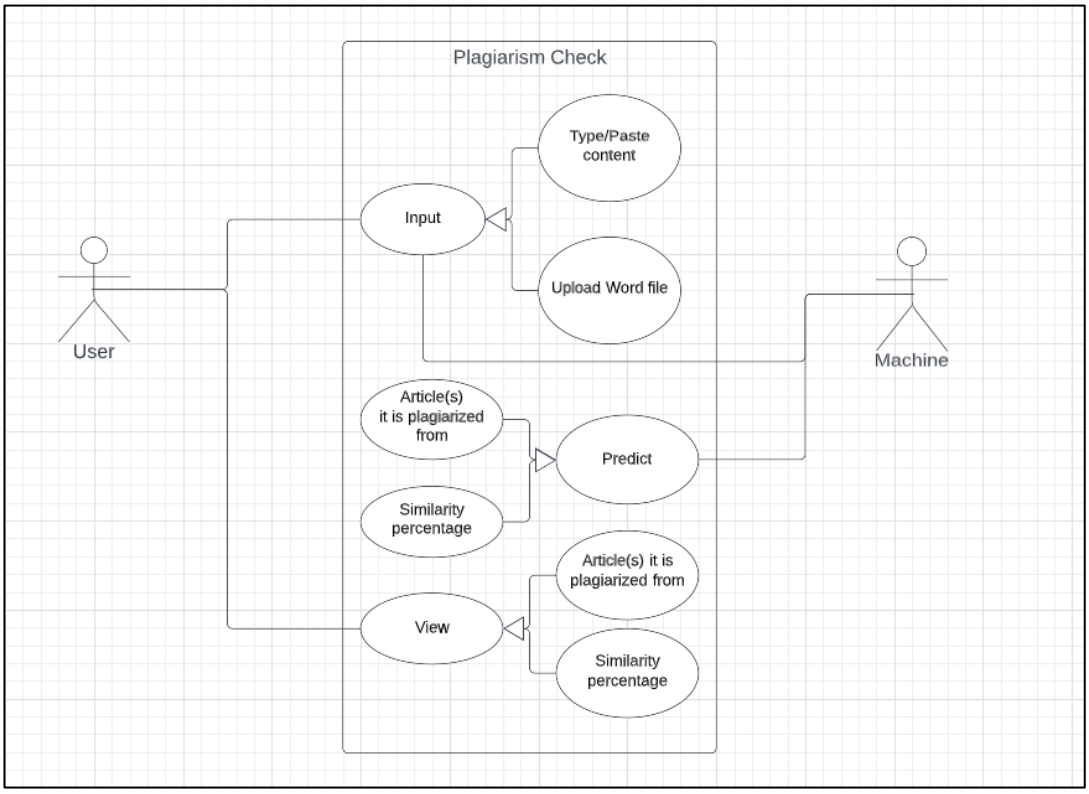

Figure 1: Use Case Diagram

4.1.2 CLASS DIAGRAM

A class diagram is a Structure Diagram whose primary purpose is to identify the potential classes in a system, their attributes, operations that each class has to perform, and how the classes in the system are connected to one another, i.e., identifying the relation between various classes. In our proposed system, we have two classes: User and Machine. The User has only one attribute, i.e., Input, and the operations that they can perform are to give input, view the plagiarized percentage and content after evaluation. Coming to Machine class, its attributes are plagiarized percentage and content. The operations that our machine can perform are creating metadata (Vector Search Index, Aka Vector Database), performing similarity checks, return plagiarized percentage and the plagiarized content.

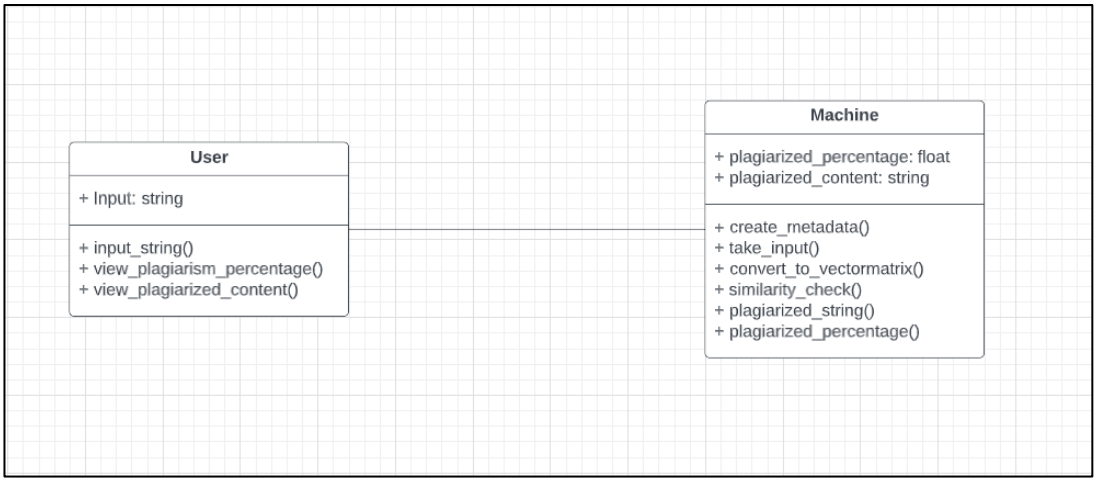

Figure 2: Class Diagram

4.1.3 ACTIVITY DIAGRAM

An activity Diagram is another Behavioral Diagram whose primary purpose is to describe the system's workflow. The website begins with text input from the user, which goes into the model which converts the input into a vector matrix. Based on this vector matrix, the model will query the database for similarity. Once the querying is done, the result is returned to the user for view.

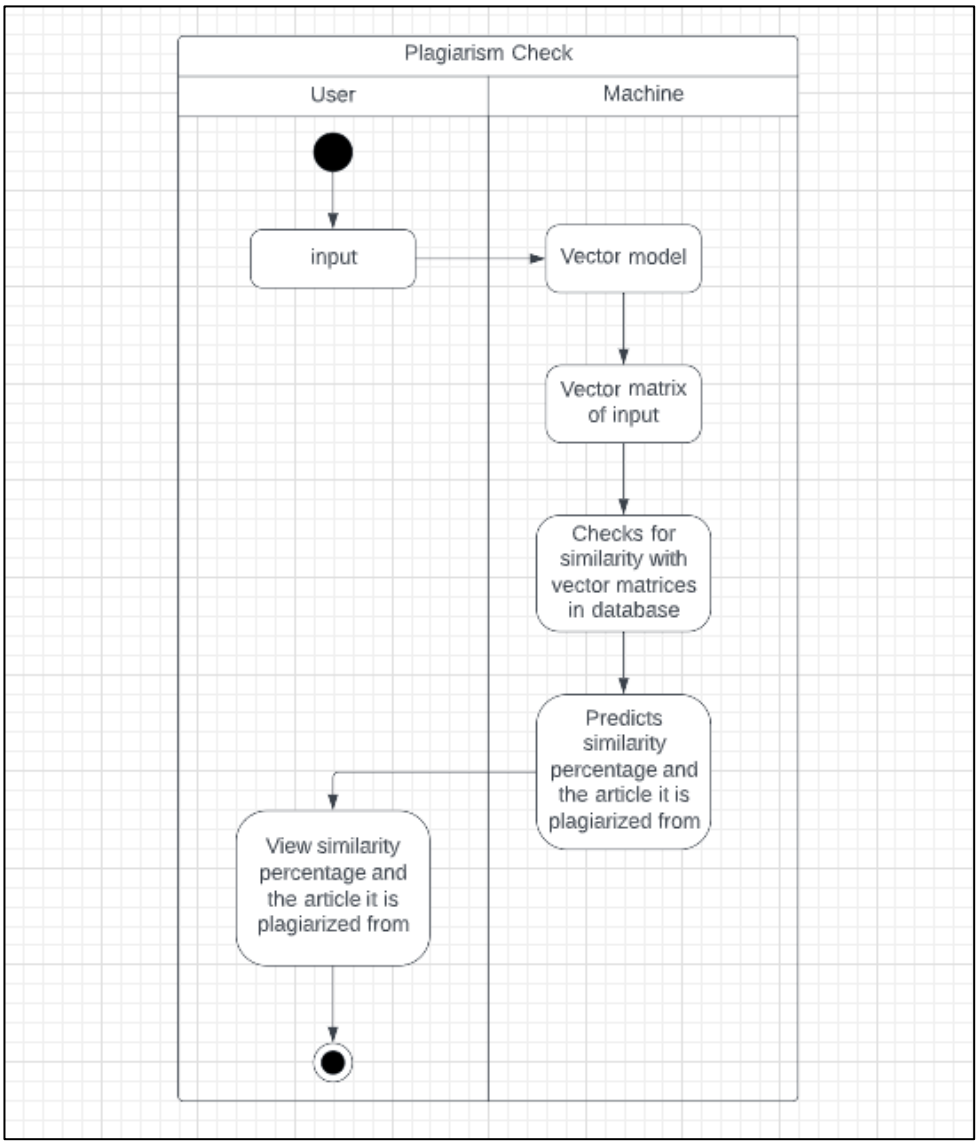

Figure 3: Activity Diagram

4.1.4 SEQUENCE DIAGRAM

A sequence Diagram is another Behavioral Diagram whose primary purpose is to describe how the different objects in the system interact. In most cases, a sequence diagram and an activity diagram are very similar to other. There are two objects in our proposed model: User and Machine. The website begins with text input from the user, which goes into the model which converts the input into a vector matrix. Based on this vector matrix, the model will query the database for similarity. Once the querying is done, the result is returned to the user for view.

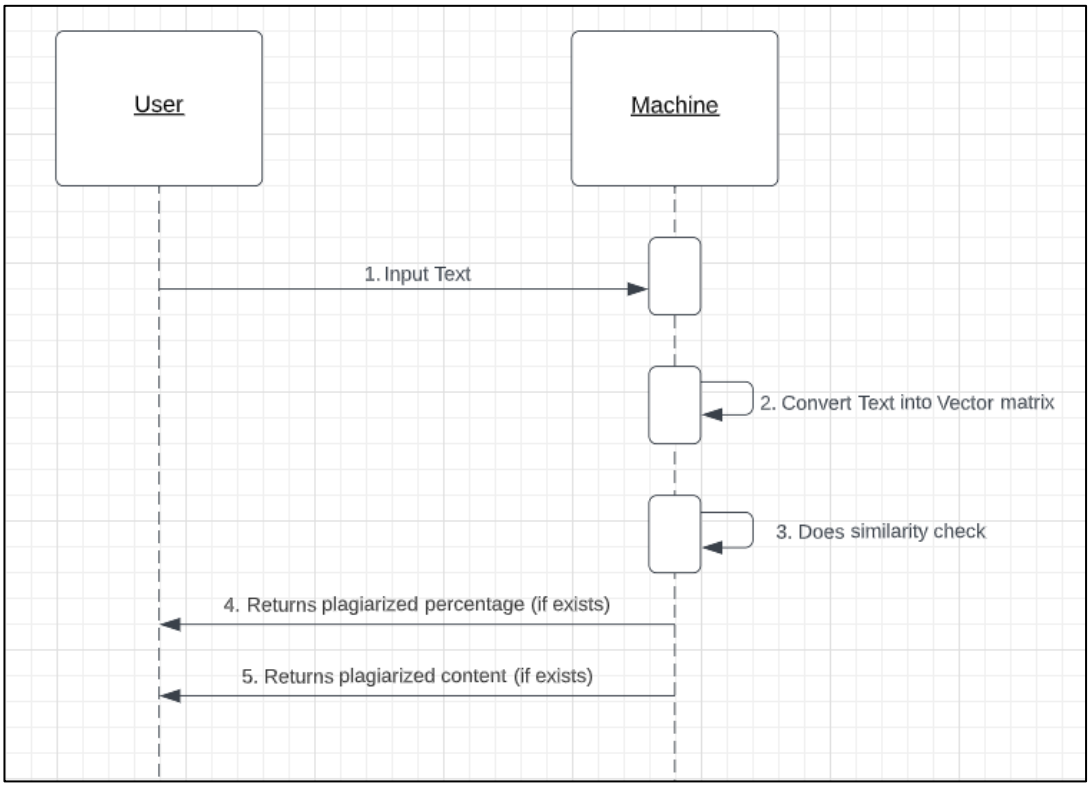

Figure 4: Sequence Diagram

4.1 MODEL OVERVIEW

The proposed Plagiarism Checker model development begins with collecting a database that will be used for training and testing our model. We need to create a vector database for the original database, i.e., a database that contains the original text and its associated vector embeddings. This vector database is given as an input to the embedding model. Next is to create indexes for these vector embeddings and store them in the vector database, which is done by the embedding model. Now, coming to querying the model when an input text is given in the website, the website will request a vector search to the embedding model; the embedding model will create an index vector for the input text and then search for similarities with index vectors already present in the vector database. Once the similarity search is completed, the similarity search results will be returned to the website for the user to view.

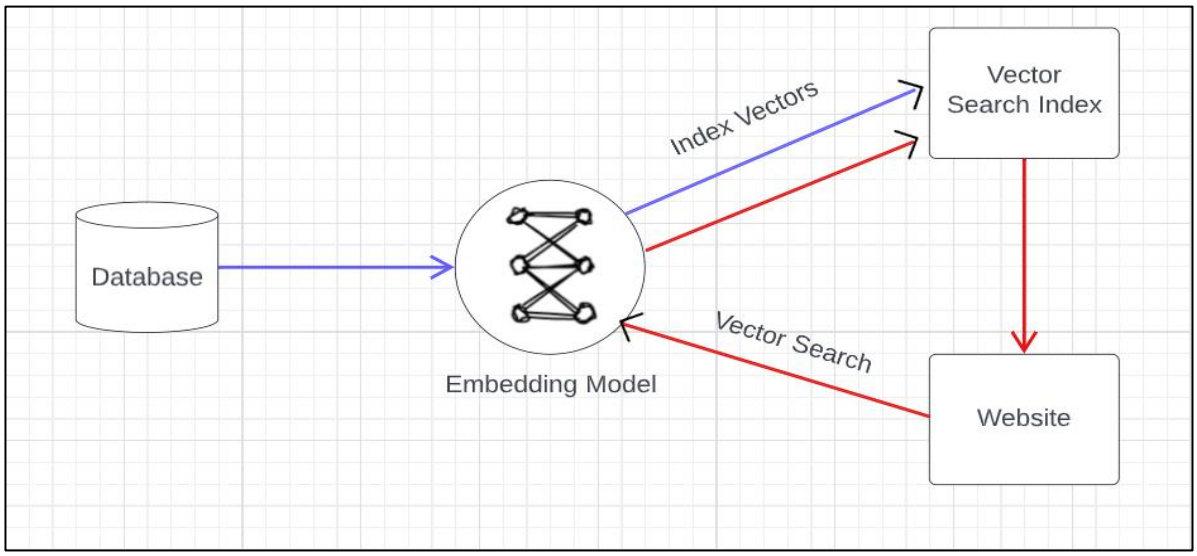

Figure 5: Model Overview

Figure 6: Embedding Model

5 OVERVIEW OF TECHNOLOGIES

5.1. Machine Learning:

One of the subsets of Artificial intelligence is Machine Learning. Machine learning enables a machine to think like a human. It performs the same task repeatedly to produce results with more accuracy. It allows machines to learn automatically and improve from experience without explicitly programming.

5.2. Natural language processing:

Natural language processing (NLP) refers back to the department of computer science— specifically, the department of AI concerned with giving computer systems the capacity to understand textual content and spoken phrases in the same manner humans can.

5.3. Vector Embedding:

Vector embedding is a concept where textual content data is converted into a listing of numbers, thus enabling the execution of various operations on the transformed data. Vectors are beneficial because when we present real-world entities like audio, images, textual content, etc., as vector embeddings, the semantic similarity among those entities may be quantified by how similar they're to every other as points in vectors. Models are skilled in translating entities into vectors.

5.4. Similarity Search:

Vector search represents photographs or bits of textual content as vectors or embeddings. They regularly assist train machine, learning models. Closeness represents more vector similarity among the embeddings, while extra distance means fewer common characteristics. One of the crucial components of efficient search is flexibility. A similarity search may be used to examine records quickly.

5.5 Vector Search Index:

A Vector Search Index, aka vector database, has features like CRUD operations, metadata filtering, horizontal scaling, indexes, and saves vector embeddings for quick retrieval and similarity searches. The way that vector databases arrange the vector embeddings allows us to compare any vector to another vector or the vector of a search query. Vector databases are excellent in "vector search" or similarity checking. Typical applications for vector search include: Search semantics, Similarity searches for unstructured data such as JSON, pictures, audio,

and video, Detection of anomalies, Record matching, and deduplication.

5.6 HTML:

HTML is a high-level language that defines the structure of content. It's a set of foundations used to enclose and separate different corridors of content to make them appear or behave in a certain way. Surrounding markers can link words and images differently, emphasize words, and make fonts taller or lower. So, using HTML, we can define paragraphs, headers, images, links, and much more so that the browser will know how to structure the web page we are looking at.

5.7 CSS:

CSS stands for Cascading Style Sheets. This language is used to style different HTML documents. It also describes and shows how the HTML elements must be presented or displayed. The CSS creates a very tremendous and significant experience. CSS is used by more than 90% of websites. The CSS is useful because it avoids duplication and makes maintenance easier.

5.8 Java Script:

It is a widely used language for web development. JavaScript is easy to understand and use because it is integrated with HTML. It is also integrated with JAVA. Its implementation allows client-side scripts to communicate with users and create various unique and effective pages. It is an interpreted programming language with OOP features.

5.9 Flask:

A Python-based micro-web framework is called Flask. Since it doesn't require specific tools or libraries, it is categorized as a micro-framework. While many already existing third-party libraries provide many components, such as functions, standards, a database abstraction layer, and other components, Flask lacks these. However, Flask allows for extensions that may be used to add application functionalities just like they were built into the core of Flask. There are extensions for object-relational mappers, form validation, upload handling, several open authentication protocols, and several utilities associated with popular frameworks.

6 REFERENCES

A. JOURNALS/ARTICLES

[1] Hiten Chavan, Mohd. Taufik, Rutuja Kadave, Nikita Chandra. "Plagiarism Detector Using Machine Learning," Department of Information Technology, Bharati Vidyapeeth College of Engineering, Navi Mumbai, India. International Journal of Research in Engineering, Science and Management Volume 4, Issue 4, April 2021.

[2] El Mostafa Hambi, Faouzia Benabbou. "A New Online Plagiarism Detection System based on Deep Learning," (IJACSA) International Journal of Advanced Computer Science and Applications Vol. 11, No. 9, 2020.

[3] Nishesh Awale, Mitesh Pandey, Anish Dulal, Bibek Timsina. "Plagiarism Detection in Programming Assignments using Machine Learning," Department of Electronics and Computer Engineering, Pulchowk Campus, Lalitpur, Nepal. Journal of Artificial Intelligence and Capsule Networks (2020). Vol.02/ No. 03.

[4] Babitha, Harshitha, Hindumathi, Reshma Farhin. Adhiyamaan. "ONLINE ASSIGNMENT PLAGIARISM CHECKER USING MACHINE LEARNING," College of Engineering (Autonomous), Hosur, India. International Journal of Advanced Research in Computer and Communication Engineering Impact Factor 7.39 | Vol. 11, Issue 4, April 2022.

B. e-WEBSITES:

[5] Natural Language Processing, *https://www.ibm.com/cloud/learn/natural-language-processing*

[6] What are Vectors Embeddings, *https://www.pinecone.io/learn/vector-embeddings/*

[7] What is Similarity Search, *https://www.pinecone.io/learn/what-is-similarity-search/*

[8] Vector Search Index, *https://www.pinecone.io/learn/vector-database/*

[9] HTML & CSS, *https://www.google.com/amp/s/www.themuse.com/amp/advice/9-reasons-every-professional-should-know-a-little-html-and-css*, *https://www.w3schools.com/css/css_intro.asp*

[10] Flask, *https://en.m.wikipedia.org/wiki/Flask_(web_framework)*