International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 07 Issue: 05 | May - 2023 Impact Factor: 8.176 ISSN: 2582-3930

AI POWERED RECASTER

RAGUL P, KARNIKA V, SIVA G, SATHYA BALAMURUGAN R
DEPARTMENT OF INFORMATION TECHNOLOGY
BACHELOR OF TECHNOLOGY
SNS COLLEGE OF TECHNOLOGY, COIMBATORE-35
(AN AUTONOMOUS INSTITUTION)

ABSTRACT

The main goal of this research is to use GPT-3 in AI to rewrite and display information according to a keyword. A platform on the web where may rephrase the provided text with appropriate words and move about a paragraph or piece of content. This is the current situation, however the content generator wasn't using the term. In order to provide material for composing emails, letters, or professional usage, we built this Recaster. Creating a platform that employs artificial intelligence to check for grammatical errors, repetitions, and the alignment of words, phrases, or clauses in the proposed system. Neural networks and artificial intelligence collaborate to reinterpret and modify the sentences. develop a website for phrases and content recasting. The suggested method will produce material for the user utilising Turnitin and GPT-3 in addition to rephrasing it.

Neural network approach instructs computers to analyse data in a manner modelled after the way the human brain does so. As a result, the AI scans the data and produces the data according to the userentered keyword.

Keywords

Plagiarism, Turnitin, research writing Natural Language Processing, GPT-3, GPT-2, Language Models, Content Generation, and artificial intelligence.

CHAPTER 1
INTRODUCTION

1.1. OVERVIEW

The World Wide Web's accessibility and low cost have made plagiarism a rising worry for educators, particularly at the tertiary level where many assignments need for original work. Academics have battled and condemned plagiarism, whether it is deliberate or not. Over the past two decades, a variety of plagiarism detection technologies has been created in response to this concern with academic integrity. Turnitin is arguably the most well-known of them up to this point, being used by over 30 million students at 15,000 universities across 150 countries. There has been a lot of study done on Turnitin since its inception regarding how well it works to identify plagiarism and stop it through. But other people have only lately started using it. The insurance industry is one of the first industries invested in innovation, the latest technology and artificial intelligence (AI). In today's world, when the rate of car accidents is increasing, car insurance companies waste millions of dollars annually, due to claims leakage. The sense of AI technology based on machine learning and deep learning can help problems such as analysing and processing data, detecting frauds, lessening risks and automating claim process in insurance industries. So, insurance firms have looked for faster damage assessment and agreement of claims. However, a development of modern applications to overcome such problems is still challenging, especially in applying deep learning for car damage assessment. Deep learning is an efficient approach for solving complex tasks, but it needs more resources for model development, i.e., for training a model, deep learning requires a huge dataset and takes more computation time. To realize deep learning approach for car damaged assessment, this paper focuses on two challenges for creating an efficient model.

These institutions located in the Middle East where the culture of research is still evolving. One of the universities that recently used the programme as part of its anti-plagiarism policy was Al Ain University. Therefore, it has been imperative to educate students not only for the software itself but also for the set of values and abilities that academic integrity needs in order to prevent a predicted state of confusion on their side. In this situation, it is quite difficult for teachers to begin educating pupils about academic integrity principles at the university level. On the one hand, some would argue whether organisations in this setting have a strong case for implementing Turnitin, about CNN models to detect damaged part with many techniques via transfer learning.

Pre-trained models are models that have already been trained on a large dataset. This allows them to be used for tasks where it would be difficult to train a model from scratch. A pre-trained model may not be 100% accurate, but it saves you from reinventing the wheel, saving time, and improving performance.

1.2. PROBLEM WITH CURRENT SCENARIO

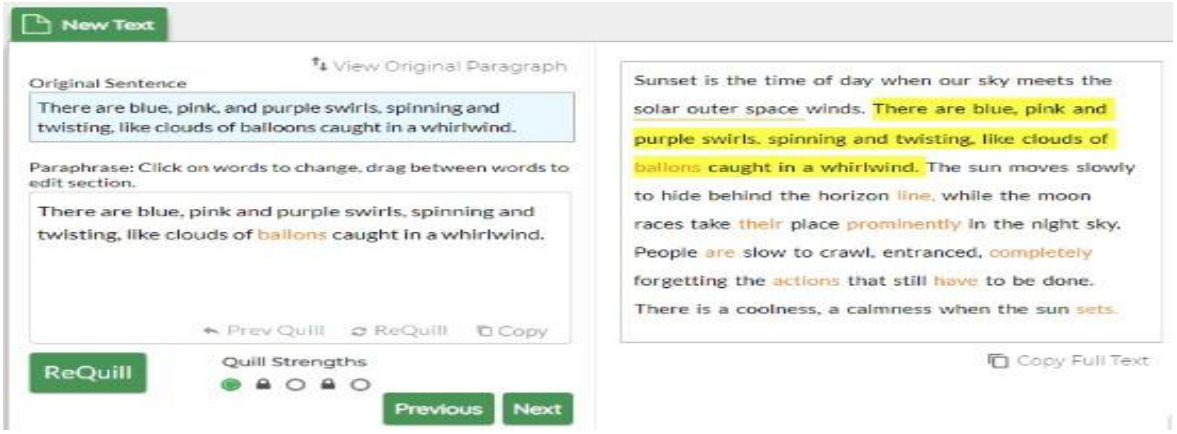

FIG : 1.2 .1 TEXT REPHRASE

1. Paraphrasing:

The ability to paraphrase involves having pupils put ideas, meanings, and information from different formats into their own words. In order for the kids to improve their writing abilities, paragraphing is crucial. Thus, this ability is the finest methods for fully comprehending the sources and incorporating them into their own work. After that, students will simply build their own writing.

The problem with the current scenario of paraphrasing and English text is that existing tools often fail to produce accurate and grammatically correct rewrites. These tools rely on algorithms that substitute words and phrases without considering the context or meaning of the original text. As a result, the output can be confusing or inaccurate, which can lead to unintentional plagiarism or a decrease in readability.

Moreover, the tools may not be able to handle certain types of text, such as technical jargon or specialized vocabulary. This can limit their usefulness for writers and researchers in specific fields.

2. English Text:

The Oxford Dictionary (2011) defines text as any written stuff. A magazine or book may contain it. Here, the term "English text" refers to academically authored materials in English. It appears to be a sentence

or paragraph that the author gave the students to use in this research. The author will use an English text from Oshima & Hogue's "Writing Academic English" for this research (1998).

Another issue is that paraphrasing tools are often used to circumvent plagiarism detection software, which can have negative consequences for academic integrity. When used improperly, paraphrasing tools can allow students to submit work that is not their own, which undermines the learning process and devalues academic achievements.

Furthermore, the use of paraphrasing tools can discourage the development of critical thinking and writing skills, as users rely on the tool to generate content rather than engaging in the creative and analytical process of writing.

In summary, the problem with the current scenario of paraphrasing and English text is that existing tools are often inaccurate and can be used to circumvent plagiarism detection, which can undermine academic integrity and discourage critical thinking and writing skills. There is a need for improved tools that can accurately and effectively assist writers and researchers in generating original content.

1.3 NEED OF RECASTER

1. Creating a website to help students and employers improve their English language skills using AI.
2. Many of them struggle to write content that is free of grammar mistakes.
3. By utilizing this AI, grammatical errors and awkward sentence constructions are reduced in the text assembled in grammatical order.
4. The student will converse with the AI robot and correct any errors.

FIG : 1.3.1 API CONNECT WITH AI

CHAPTER 2

LITERATURE SURVEY

"GTP-3 Powered System for Content Generation and Transformation" by **Shruti Saravanan** and **K. Sudha** Enabling computer systems to understand and generate natural language has been an up-and coming field of research. Latest advancements in Natural Language Processing (NLP) have made headway progress in facilitating this, like the GPT-3 language prediction model created by AI. Given the capacity of the GPT-3 model, this study capitalizes on how the model can be used to generate and transform content without manual help from humans – how well plausibly GPT3-authored text most nearly passes as human-like prompts for content generation and manipulation purposes. This attempt is presented in the context of automated story writing. It also sheds light on the potential abuses of the tool and its raw capabilities as its limitations.

"Research on Error Detection Technology of English Writing Based on Recurrent Neural Network" by **Wang Wei**, **Li Yong-An**, **Ma Lun** and **Qu Qianqian** In learning English, learners are faced with various challenges, such as listening, speaking, reading and writing. Among them, writing is the most important and difficult one, and grammatical errors are the most common type of errors in English writing. The research and implementation of grammatical error detection and correction in English writing is of great significance to both English learners and English teachers. In this paper, we propose a sequence annotation model based on recurrent neural network to solve the problem of the influence of grammar errors and other noises in English writing corpus on sequence information extraction. Aiming at the grammatical errors in English writing, this paper proposes two methods of English grammatical error detection and correction. The experimental results show that the accuracy of RNN is higher than that of CAMB after 10 to 20 iterations, ranging from 0.848 to 0.916.

"Paraphrasing: An Effective Comprehension Strategy" by **Sharon B. Kletzien** Paraphrasing, somewhat different from retelling and summarizing, helps students monitor their understanding and incorporate new knowledge with what they already know about a topic. Paraphrasing helps students realize that comprehension is the goal of reading.

"Turnitin: The student perspective on using plagiarism detection software" by **Stephan Dahl** Recently there has been an increasing interest in plagiarism detection systems, such as the web-based Turnitin system. However, no study has so far tried to look at how students react towards those systems being used. This exploratory study examines the attitudes of students on a postgraduate module after using Turnitin as their standard way of submitting work and getting feedback. Overall, students reacted positively towards the system. However, the study also found evidence of a group of students who were less positive, which seemed to be a result of their insecurity about how to quote correctly.

Advantage and Disadvantage of Recaster

- Demonstrating understanding: When you paraphrase, you show that you have understood and can explain the ideas in your own words. This can be especially useful in academic settings, where demonstrating understanding is important.
- Expanding your vocabulary: Paraphrasing can help you learn new words and phrases, as you may need to find different ways to express the same idea.
- Paraphrasing can take a significant amount of time, especially if the original text is lengthy or complex.
- It can be difficult to accurately paraphrase without introducing errors or misunderstandings.
- Some people may use paraphrasing as a way to avoid plagiarism, but it is still possible to commit plagiarism through paraphrasing if you do not properly cite the original source.

Summary

- A paraphraser is a tool or person that rewrites or restates a piece of text in a different way while maintaining the original meaning.
- It is often used to avoid plagiarism or to clarify complex language.
- Paraphrasing can be done manually by rewording the text or using software or online tools to automatically generate a paraphrased version.
- It is important to ensure that the paraphrased text is not too similar to the original, as this can still be considered plagiarism.

CHAPTER 3
EXISTING SYSTEM

3.1 OVERVIEW

There is currently no suitable platform for recasting sentences, and those that do exist may not do it perfectly. The time taken is long and the precision is quite poor. You're not alone if the term "artificial intelligence" makes you think of Hollywood robots and lab rats (and you have a stellar imagination and are probably super cool).

3.2 EXISTING SYSTEM

A paraphrasing tool is used to rewrite or rephrase a sentence without altering its meaning. This is accomplished by substituting any number of alternate versions for specific words, phrases, sentences, or even whole paragraphs to create a slightly different variant.

- Low Accuracy
- High Complication

The current system of paraphrase tools is made to help people rephrase or rewrite words and paragraphs in order to avoid plagiarism or to make text easier to read. Those who need to produce original content or prevent plagiarism in their work, such as writers, researchers, and students, frequently use these tools.

The current approach, meanwhile, has several drawbacks. The poor accuracy of many paraphrasing technologies is a significant problem since it might lead to awkward or grammatically incorrect phrases that do not effectively express the intended meaning. Due to the tool's potential to create language that is overly close to the original source, inaccurate paraphrasing might also result in inadvertent plagiarism.

The high degree of complexity associated with employing a variety of paraphrase tools is another drawback of the current approach. Some devices need a either a difficult installation process or substantial training are required for use. In certain cases, users must manually enter the text to be paraphrased or negotiate a challenging user interface. Some users may find it challenging to take use of the advantages of paraphrase tools due to these entrance obstacles.

In conclusion, the current system of paraphrasing tools has several important shortcomings in terms of accuracy and usability, even though it might be effective for creating unique material and preventing plagiarism. Future developments of these tools should concentrate on resolving these problems, for instance by simplifying the user interface and creating more precise algorithms.

Methodology: Turnitin

Turnitin is a plagiarism detection tool that can help identify instances of copying or paraphrasing in written work. By using Turnitin, teachers and instructors can check students' work for potential plagiarism and ensure that their writing is original and properly cited. This can help promote academic integrity and prevent students from accidentally or intentionally committing plagiarism. Additionally, Turnitin can provide feedback to students on how to improve their writing and avoid plagiarism in the future.

Turnitin also provides a feedback studio, a tool that allows instructors to provide feedback on student work. Instructors can add comments, suggestions, and corrections directly to the text of the submitted work using a variety of tools, including a grading rubric. This tool helps instructors assess the quality of student work and provide actionable feedback to improve student performance.

Overall, Turnitin is a valuable tool for promoting academic integrity and improving student writing skills. It allows instructors to identify potential instances of plagiarism and provide targeted feedback to students, helping to improve student learning outcomes and promote a culture of academic integrity in educational institutions.

Advantage

- Through the detection of instances of plagiarism in written material, it can support academic integrity.
- They can quickly and simply assess students' papers for possible plagiarism using Turnitin, which can help them spot any issues and fix them before they get worse.

Disadvantage

- Turnitin might not be able to identify plagiarism from sources like unpublished documents or private communications that are difficult to find or search.

- In certain situations, it can be the teacher's or the instructor's responsibility to carefully check the work and find any instances of plagiarism.

Summary

- Read and Make Notes. Carefully read the text that you want to paraphrase.
- Find Different Terms. Find equivalent words or phrases (synonyms) to use in place of the ones that you have picked out.
- Put the Text into Your Own Words.
- Check Your Work.

PROPOSED SYSTEM

3.3 OVERVIEW

This study major objective is to rewrite and display information using GPT-3 in AI in accordance with a keyword. a website where you can change the provided text's wording and move around a paragraph or piece of content. Although the term was not used by the content generator, this is the current circumstance. We created this Recaster to offer content for drafting emails, letters, or other formal correspondence. constructing a platform that uses artificial intelligence to check the proposed system for grammatical mistakes, repetitions, and the alignment of words, phrases, or clauses. Artificial intelligence and neural networks work together to reinterpret and change the sentences. To recast language and information, we create a website. The suggested approach will result in.

3.4 PROPOSED SYSTEM

- We are developing a platform that utilize artificial intelligence to check for grammatical faults, redundancies, and the alignment of words, phrases, or clauses.
- Artificial intelligence and neural networks work together to recast and manipulate the sentences.

The following modules might be part of a GPT-3 system proposal:

Module for Input:

This module is in charge of taking user-provided input text. It can be a text editor where the user enters or pastes the text to be created or paraphrased.

Before the input text is sent on to the GPT-3 model, this module is in charge of preparing it. Preprocessing activities might include tokenization, stop-word removal, and other data cleaning procedures.

The system's central component, the GPT-3 Model Module, generates or recasts the text based on the input text and the given parameters. GPT-3 is the best option since it is a deep learning model that can produce language that sounds like human speech.

Methodology: GPT-3

GPT-3, or Generative Pretrained Transformer 3, is a state-of-the-art language processing system developed by AI. It can generate human-like text and can be used for a variety of natural language processing tasks, such as machine translation, summarization, and text generation. In a proposed system, GPT-3 could potentially be used to improve the accuracy and efficiency of language processing tasks, such as identifying instances of plagiarism in written work. It could also be used to provide personalized feedback and guidance to students on how to improve their writing and avoid plagiarism.

Determine the Use Case: First, determine the use case for GPT-3. What problem are you trying to solve, and how will GPT-3 help you solve it? Some common use cases include chatbots, language translation, content creation, and language understanding.

Choose a Programming Language: GPT-3 supports several programming languages, including Python, Java, and C++. Choose a language that is compatible with your development environment and team's skill set.

Choose an API: GPT-3 provides several APIs for developers to use, including the Open AI API and GPT-3 Playground. The API you choose will depend on your use case and development needs. Obtain API Keys: Once you have chosen an API, you will need to obtain API keys from the provider. These keys will give you access to the API and allow you to integrate GPT-3 into your application.

Develop and Test: Develop your application using GPT-3, making sure to test your code thoroughly to ensure that it functions as intended.

Deploy and Monitor: Once your application is developed, deploy it to your desired platform, and monitor it to ensure that it is performing as expected. Be sure to monitor API usage and costs to avoid exceeding your budget. Stay Up-to-Date: Finally, stay up-to-date with the latest developments and updates from the GPT-3 provider, as well as best practices for using the technology.

Advantage

- It is suitable for a range of tasks involving natural language processing because of its capacity to produce text that resembles human speech**.**
- GPT-3 can comprehend and produce text similarly to how a person would since it has been educated on a vast quantity of data.
- GPT-3 can continue to develop and become even more efficient over time since it can learn from new information and tasks and adapt to them.

Disadvantage

- For certain applications, this might make it challenging and expensive to utilise, especially for those with limited resources or finance.
- The sheer size and complexity of GPT-3 may restrict its applicability for specific jobs since some users may find it challenging to comprehend and utilise efficiently.

DATAFLOW DIAGRAM

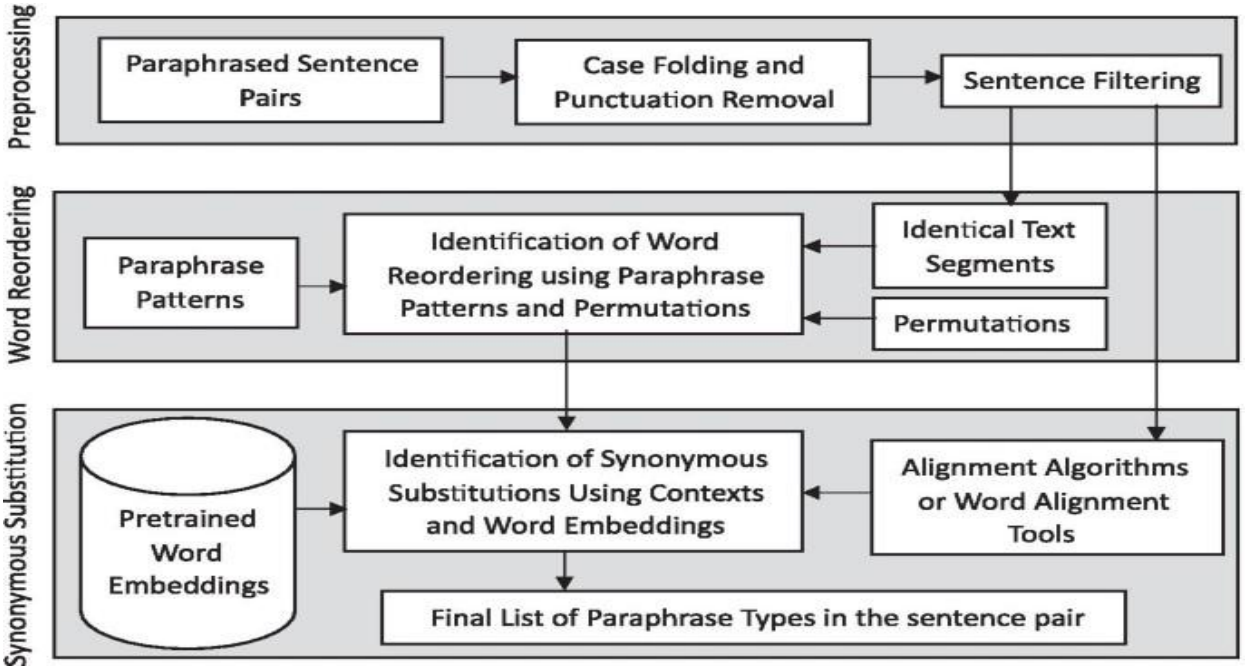

FIG 3.5: LEVEL 1 DFD

CHAPTER 4
SYSTEM REQUIREMENTS

Hardware Requirement:

- Operating system : Windows & Linux
- System type : 64 bit Operating system
- Processor : Intel core i5
- RAM : 4 GB

System Requirements:

- Operating system : Windows & Linux
- Web Browser : Chrome, Mozilla fire fox
- Front End : HTML, CSS
- Back End : JavaScript

Software Description:

Software environment is a technical specification of requirement of software product. This specifies the environment for development, operation and maintenance of the product. A software description provides a thorough breakdown of a software product, including all of its features, functions, and advantages. It contains details on the functionality of the programme as well as instructions on how to utilise it for the target market.

The aim and advantages of the programme should be briefly summarised in a decent software description, together with any technical information about how the product works and the prerequisites for operating it.

Technology Used:

- HTTP Basics
- HTML
- CSS
- JavaScript

HTTP BASICS:

An HTTP client sends a request message to an HTTP server. The server, in turn, returns a response message. In other words, HTTP is a pull protocol, the client pulls information from the server (instead of server pushes information down to the client). HTTP is a stateless protocol.

The protocol used for communication between web servers and web clients (such as web browsers) is called HTTP, or Hypertext Transfer Protocol. The client sends a request to the server, and the server responds by sending the client a response; this is known as a request-response protocol.
The following are some of the fundamental ideas and elements of HTTP:

URL (Uniform Resource Locator):

A URL (Uniform Resource Locator) is a string that indicates where on the internet to find a particular resource, such as a webpage, picture, or video. The hostname (such as www.example.com), the protocol (such as HTTP or HTTPS), and the route to the resource are some of its constituent parts.

HTTP Methods:

HTTP includes a number of verbs or methods that specify the kind of action being taken on the resource defined by the URL. The four most often used techniques are GET (used to get a resource), POST (used to submit data to a server), PUT (used to update a resource), and DELETE (used to delete a resource).

HTTP Headers:

HTTP headers are pieces of metadata that are attached to a request or response and offer further details about the message. For instance, the User-Agent header identifies the client initiating the request, whereas the Content-Type header defines the type of the material being transmitted or received.

HTTP Status Codes:

Three-digit numbers known as HTTP status codes are used to describe a request's or response's current state. Success status code 2xx is the most prevalent. 3xx (redirect), 4xx (client error), and 5xx (server error).

HTTP Cookies:

The server saves a few bytes of information known as HTTP cookies on the client's machine. They are used to store data such user preferences, login information, and the contents of shopping carts.

Overall, HTTP is a core protocol for the web, and creating online applications and using web APIs requires a comprehension of its fundamental ideas and elements.

HTML:

HTML (HyperText Markup Language) is the code that is used to structure a web page and its content. For example, content could be structured within a set of paragraphs, a list of bulleted points, or using images and data tables.

The most recent version of HTML, known as HTML5, is used to build online pages and web applications. The World Wide Web's foundational technology, it offers a standardised way to define the organisation and content of web pages. Here are some essential HTML5 characteristics and ideas:

Semantic components:

HTML5 comes with a tonne of brand-new semantic components that help web designers better organise their websites. A single piece of content can be defined via the element, which can also be used to define the top portion of a page.

Support for multimedia:

HTML5 comes with new components and APIs for integrating audio and video files directly into web pages. The Media Capture API makes it possible to capture audio and video from mobile devices, and the video> and audio> components make it simple for developers to add media files in their web pages.

Canva:

HTML5 adds a new element called "canvas" that allows users to use JavaScript to create animations, images, and other visual effects right in the browser.

Responsive web development:

Web page development that can adjust to various screen sizes and devices is now possible thanks to HTML5's improved support for responsive web design. Features like media queries, which enable styles to be applied based on the size, help achieve this media query.

Form Enhancements:

HTML5 comes with new input types and features that make creating and validating forms simpler. The new input type="date"> and input type="time"> elements, for instance, make it simple to enter dates and times, and the necessary property may be used to make sure that certain form fields are filled out.

All in all, HTML5 offers a number of new capabilities and enhancements that make it simpler to develop rich, interactive online apps that function flawlessly across various devices.

CSS:

CSS is the language for describing the presentation of Web pages, including colors, layout, and fonts. It allows one to adapt the presentation to different types of devices, such as large screens, small screens, or printers. CSS is independent of HTML and can be used with any XML-based markup language.

The most recent version of the CSS specification, known as CSS3 (Cascading Style Sheets level 3), is used to style web pages. It offers a standardised approach to define the layout, colours, fonts, and animations used in the presentation of web content. Here are some essential CSS3 characteristics and ideas:

Selectors:

CSS3 adds new selectors that make it simpler for developers to focus on particular components on a page. For instance, targeting items based on their placement inside a parent container is possible using the:nth-child() selector.

Responsive Design:

Development of web pages that can adjust to various screen sizes and devices is made possible by CSS3's improved support for responsive web design. Features like media queries, which enable styles to be applied based on the size of an element, help achieve this on this screen.

Transitions and Animations:

CSS3 offers new properties and syntax that may be used to create transitions and animations without the need of JavaScript, right in the browser. While the animation property enables the creation of intricate animations, the transition property enables seamless transitions between various states of an element.

Flexbox:

A new layout module called Flexbox is part of CSS3, and it offers a flexible method for organising items on a website. Flexbox makes it simple for developers to design layouts that can change to fit various screen sizes and device orientations.

Grid:

Another new layout module in CSS3 is called Grid, and it offers a strong method for making intricate grid-based layouts. Developers may use Grid to make multi-column layouts that can adjust to various screen sizes and device orientations

JavaScript:

JavaScript is used by programmers across the world to create dynamic and interactive web content like applications and browsers. JavaScript is so popular that it's the most used programming language in the world, used as a client-side programming language by 97.0% of all websites.

JavaScript is a flexible programming language that may be applied to a variety of web development tasks. Here are some examples of sophisticated JavaScript web page usages:

Dynamic Web Content:

JavaScript may be used to produce dynamic web content, which is material that changes on a web page in response to user input or other events. For instance, JavaScript may be used to update a web page's content without requiring a complete page refresh, improving the user experience and responsiveness.

User Interface Enhancement:

By adding interactive components like dropdown menus, sliders, and pop-up dialogue boxes, JavaScript may be used to improve the user interface of a web page. These elements may be made by developers using JavaScript, resulting in a more entertaining and straightforward user experience.

AJAX:

Web pages that can asynchronously connect with a web server using JavaScript and AJAX (Asynchronous JavaScript and XML) may be created, enabling quicker and more responsive online applications. An AJAX-powered web page, for instance, can update specific sections of the page without requiring a whole page refresh.

Web APIs:

To access and modify data from third-party services like Google Maps, Twitter, and Facebook, JavaScript may be used with a variety of Web APIs (Application Programming Interfaces). Web application developers may construct more robust and integrated apps by utilising JavaScript to access these APIs.

Single-Page apps:

Instead of requiring separate HTML pages for each piece of content, single-page apps (SPAs) may be made using JavaScript. SPAs load all the essential material and functionality into a single HTML page. By minimising server queries and page load times, SPAs may offer a more responsive and fluid user experience.

In general, JavaScript is an effective tool for web developers, offering a variety of possibilities for building dynamic and interesting web applications. JavaScript is a crucial component of contemporary web development because of its adaptability and simplicity.

CHAPTER 5
IMPLEMENTATION

OVERVIEW

A rephraser is a tool or software that helps rewrite or rephrase text or sentences to improve the readability and clarity of the content. It can be used for a variety of purposes, such as avoiding plagiarism by changing the wording of a text without changing its meaning, or to simplify complex language for easier comprehension. Rephrasers often use natural language processing algorithms to analyze and understand the structure and meaning of the original text, and then generate a new version of the text that conveys the same ideas in different words. Some rephrases also include features such as synonym suggestion and grammar checking to help improve the quality and accuracy of the rephrased content.

5.1 SYSTEM IMPLEMENTATION

List of modules:

- Home page
- Text area
- Recasting
- Content genration

DATASET DESCRIPTION

Home page:

From this module user can register their account, when user create, which identifies users uniquely. The primary interface via which a user would access the system would be this module. To make it easier for the user to access different system capabilities, it could have elements like a login screen, navigation menu, and search bar.

Text area:

In this module, user can enter the text content or key word for rephrasing as well as content generation. The user might enter text that needed to be processed or changed in this module. It could come with tools for text formatting, a spell checker, and a grammar checker.

Recasting:

In this module, when the user hits the recasting button AI algorithm gets start processing in the backend. The user might change the text's organisation with this module. For instance, it could provide functions that allow the user to change the original text while retaining the original meaning, including synonym substitution, phrase rephrasing, and paraphrasing.

Content generation:

After the user enter the content, AI analysis the content then recasting the content then shows result of the rephrased content. When user enter key word AI analysis and then produce the related results. The user would be able to create new content using the input text thanks to this module. To assist the user with producing new material based on the supplied text, it could contain capabilities like text summarization, article production, and content expansion.

The user could observe the system's output thanks to this module. It could provide options for previewing the revised or created text, downloading the result in several file types, and posting the result to social media or email.

Settings:

With the help of this module, users may customise the system's settings, including the output format, language preferences, and other personalization choices.

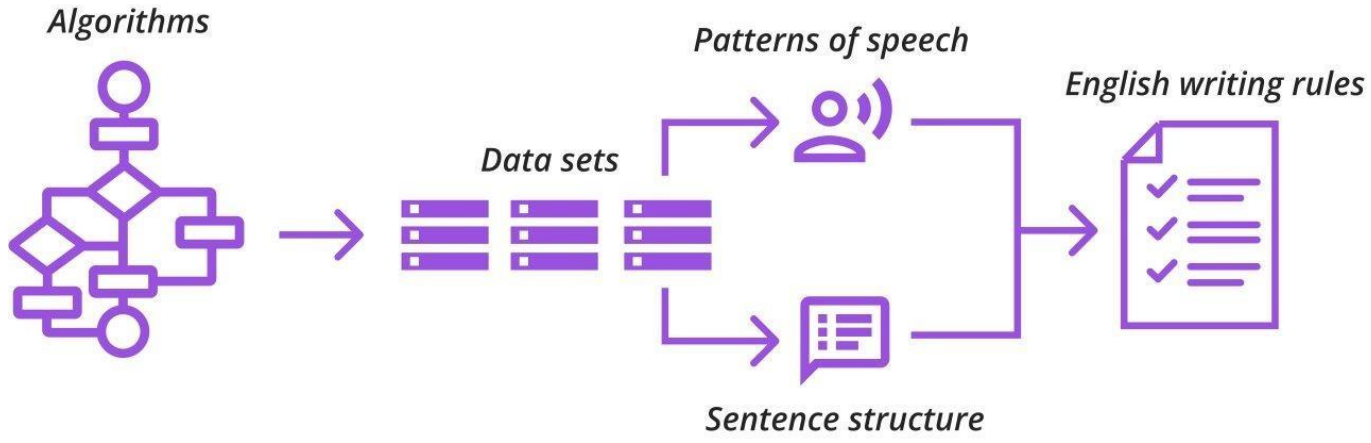

FIG:5.1.1: HOW AI LEARNS

5.2 DEFINING RECASTER USING AI

Using AI writing tools is really easy. Like, really easy. To show you just how simple this process will be for you, we're going to break it down into small steps.

- Pick your AI writer
- Pick your Topic
- Start Writing.

5.3 AI Assist

While there are certain AI writing tools available that need payment, many choices are available as part of a free plan (including, yes, Recaster Co-Writer!). With regard to our Co-Writer, the quality is present without the need for a credit card. Free tools can nevertheless be just as effective as those that cost money.

We are able to provide certain tools for free since AI Recaster has Premium subscriptions.

Not just those with money should have access to effective writing. Everyone should have access to the resources that will enable them to develop their writing abilities since everyone can be a competent writer. Our Recaster now have greater ability to manage the type of text that our AI is producing thanks to this feature. Please be aware that if you choose to exercise "maximum creativity," you may wish to spend some time fine-tuning the factual accuracy of the content that is created.

Finding the right model:

A fantastic method to learn what the API can accomplish is to experiment with Davinci. Once you know what you want to do, you can stick with Davinci if cost and speed aren't an issue, or you may switch to Curie or another model and try to optimise around its features. Use this tool to evaluate outputs, parameters, and reaction times amongst various models before downloading the data into an.xls spreadsheet.

Implementation of GPT-3:

A set of models that can understand and generate natural language. Our GPT-3 models can understand and generate natural language. We offer four main models with different levels of power suitable for different tasks. Most capable GPT-3 model. Can do any task the other models can do, often with higher quality, longer output and better instruction-following. Also supports inserting completions within text.

To generate output, GPT-3 has a very large vocabulary, which it can combine to generate sentences. These words are sorted into different categories (nouns, verbs, adjectives, etc.), and for each category, there is a "production rule", which can be used to generate a sentence. The production rules can be modified with different parameters.

There are several concerns and stages involved in implementing GPT-3. A high-level summary of the implementation procedure is given below:

Establish the Use Case: Establish the GPT-3 use case first. What issue are you attempting to resolve, and how might GPT-3 assist you? Chatbots, language translation, content generation, and language understanding are a few examples of frequent application cases. Pick a Programming Language Python, Java, and C++ are just a few of the many languages that GPT-3 supports. Pick a language that works with your team's skill set and development environment.

Select an API: GPT-3 offers a number of APIs, including the Open AI API and GPT-3 Playground, for developers to use. Depending on your use case and development requirements, you will select an API.

noun + verb = subject + verb

noun + verb + adjective = subject + verb + adjective

verb + noun = subject + verb

noun + verb + noun = subject + verb + noun

noun + noun = subject + noun

noun + verb + noun + noun = subject+ verb + noun + noun

In addition, GPT-3 is able to understand negations, as well as the use of tenses, which allows the model to generate sentences in the past, present and future.

API keys must be obtained from the supplier. You may use the API and include GPT-3 into your application with the help of these keys.

Create and Test: Create your application using GPT-3, and be careful to properly test your code to make sure it works as expected.

Once your application is complete, deploy it to the platform of your choice and keep an eye on it to make sure it is operating as expected. To stay inside your budget, keep an eye on API usage and charges.Last

but not least, keep up with the most recent advancements and upgrades from the GPT-3 supplier as well as recommended usage techniques.

Overall, careful consideration and planning are needed when implementing GPT-3 to make sure it meets your development needs.

Three Phases

- **Generative:** Generative models are a type of statistical model that are used to generate new data points. These models learn the underlying relationships between variables in a dataset in order to generate new data points similar to those in the dataset.
- **Pre-trained:** Pre-trained models are models that have already been trained on a large dataset. This allows them to be used for tasks where it would be difficult to train a model from scratch. A pretrained model may not be 100% accurate, but it saves you from reinventing the wheel, saving time, and improving performance.
- **Transformer:** A transformer model is a famous artificial neural network invented in 2017. It is a deep learning model that is designed to handle sequential data, such as text. Transformer models are often used for tasks such as machine translation and text classification.

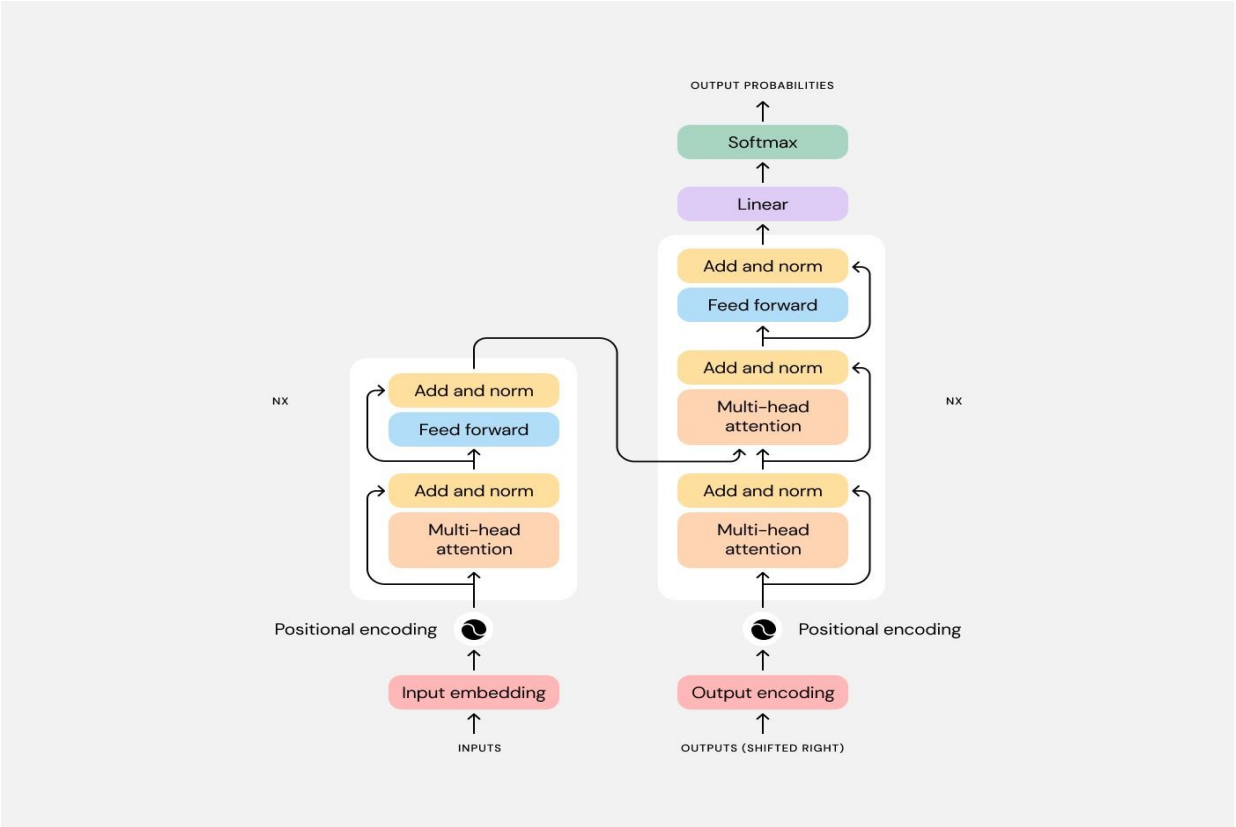

FIG:5.3.1 GPT-3 ARCHITECTURE

Turnitin:

The American firm Turnitin, LLC, a division of Advance Publications, operates the Turnitin (stylized as turnitin) online plagiarism detection tool.

It was established in 1998 and offers licences to colleges and high schools. These institutions utilise the software as a service (SaaS) website to verify submitted works against its database and the content of other websites in an effort to detect plagiarism. Results may be used in formative assessment to assist students learn to prevent plagiarism and improve their writing, as well as to detect similarities with existing sources.

Educational institutions utilise the software platform Turnitin to encourage academic honesty and combat plagiarism. It enables teachers to search a huge library of web sources, previously submitted work, and other pertinent information for textual similarities in student writing.

When students turn in their writing to Turnitin, the software creates an originality report that flags any instances of similar content and assigns a score for overall similarity. This report assists teachers in spotting possible instances of plagiarism and giving students the feedback they need to enhance their writing and uphold academic integrity.

Additionally, Turnitin offers a feedback studio, a tool that enables teachers to comment on student work. Using a number of tools, instructors can immediately add comments, recommendations, and edits to the text of the submitted work resources, such as a rubric for grading. With the use of this application, teachers may evaluate the calibre of their students' work and offer useful criticism to help them perform better.

Overall, Turnitin is a useful tool for fostering academic honesty and enhancing writing abilities in students. It enables teachers to spot possible instances of plagiarism and provide students focused criticism, improving student learning outcomes and fostering an academic integrity culture in educational institutions.

Turnitin offers two things for each piece of submitted work

- Similarity index that shows how much of the submitted paper Turnitin has determined to be similar to content from other sources.
- An originality report that provides more information about each of these matches, including the source(s), is available. These can be publications like websites, books, journals, and articles, as well as work that has already been turned in via Turnitin.

However, AI may also be utilised to facilitate cheating, as with all arms races. For instance, the GPT-3 algorithm from AI is so strong that it can produce sentences that appear to be written by a human starting with just a few basic phrases. According to one research, the articles produced by GPT-3 were almost as convincing as the original New York Times pieces (83% of respondents assessed the New York Times articles as "credible," compared to 72% for the GPT-3 samples).

AI may therefore be utilised by both sides of the playing field, just like most other things. The best strategy to combat cheating in schools may be to teach pupils that rushing through their education would only harm them in the long run. But in today's age of instant gratification, that's easier said than done.

Turnitin's algorithm examines the submitted work and compares it to a sizable database of previously submitted work, web sources, and other pertinent items to provide originality results. Any instances of matching text, such as direct quotations, paraphrases, and other instances of textual similarity, are highlighted

in the report. Additionally, the report provides a similarity score that quantifies the degree to which the submitted work resembles that of other sources.

Using the tool called Feedback Studio, teachers can give their students feedback on the assignments they've turned in. Instructors can use the tool to make direct edits, recommendations, and comments in the text of submitted work. A customizable grading rubric is also included in Feedback Studio to assist instructors evaluate student work quality and give useful giving the student feedback.

Overall, Turnitin's originality reports and comments Studio give instructors insightful information about the quality and originality of submitted work, enabling them to more accurately evaluate student learning outcomes and offer personalised comments to help students do better.

GPT-3:

A neural network machine learning model trained using internet data called GPT-3, or the third generation Generative Pre-trained Transformer, can produce any kind of text. It was created by AI, and it just needs a tiny quantity of text as an input to produce huge amounts of accurate and complex machinegenerated text.

Over 175 billion machine learning parameters make up the deep learning neural network used in GPT-3. To put things in perspective, Microsoft's Turing NLG model, which has 10 billion parameters, was the biggest trained language model prior to GPT-3. GPT-3 will be the biggest neural network ever created as of early 2021. As a result, GPT-3 is superior to all earlier models in terms of creating text that appears to have been produced by a person.

Natural language generation, which focuses on producing natural text in human language, is one of the main components of natural language processing. For robots, who don't fully comprehend the subtleties and complexity of language, producing information that is intelligible by humans is a difficulty. GPT-3 is trained to produce genuine human writing by using online text.

GPT-3 has been used to generate vast volumes of high-quality content with just a tiny amount of input text, including articles, poems, tales, news reports, and conversation.

A fresh piece of text that is relevant to the situation is automatically generated by GPT-3 in response to any text that is entered into a computer. Anything with a text structure may be created with GPT-3, and

not simply text in human languages. Additionally, it is capable of automatically producing computer code as well as written summaries.

Finishing: It is clear from the research that the open source initiatives listed - OpenAI, Hugging Face, EleutherAI, and GPT-Neo - have worked hard to offer a free and unlicensed model as a check on Microsoft's hegemony in the AI sector. Turnitin's efforts to create a human-centered AI system might benefit from the more established and sophisticated language models that OpenAI and Hugging Face have previously created. Although they are still relatively new, EleutherAI and GPT-Neo are gaining popularity in the open-source community. Turnitin may think about collaborating with Hugging Face or OpenAI to benefit from their comprehensive language models, or it might help EleutherAI and GPT-Neo in their attempts to develop effective open-source language models.

BLOCK DIAGRAM

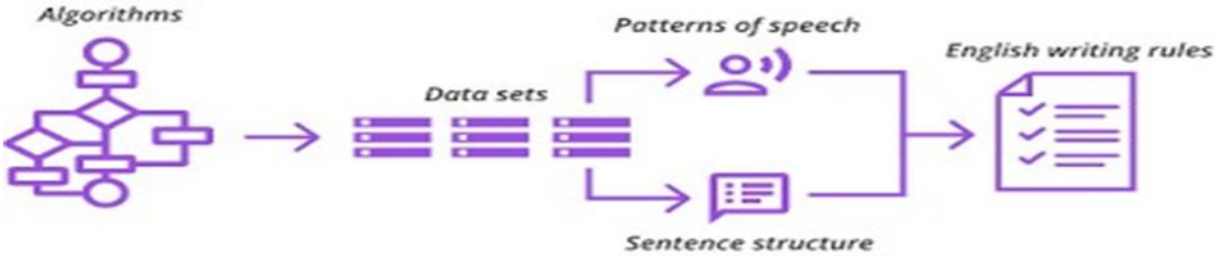

FIG: 5.3.2: AI RECASTER

As an AI agent, I am aware that the main goal is to develop even AI models that are more powerful and significant and promote a human-centered approach. The goal of Turnitin's commitment to responsible AI is to advance learning outcomes and integrity on a broad scale by assisting institutions, teachers, and students. Turnitin believes that AI should be developed with the best of intentions.

According to the sub-task, it is clear that open-source projects like OpenAI, Hugging Face, EleutherAI, and GPT-Neo have worked hard to offer a free and unlicensed model as a check on Microsoft's exclusive monopoly in the AI industry. Given Turnitin's dedication to developing AI that is human-centered, collaborating with OpenAI or Hugging Face to take advantage of their extensive language models might be

advantageous. The development of potent open-source language models by EleutherAI and GPT-Neo might instead be supported.

The goal of building human-centered AI models is aligned with a commitment to responsible AI and giving students, educators, and institutions the tools they need to make informed decisions. Turnitin can go closer to this objective by collaborating with established open-source projects or aiding fresher ones.

Prompts:

By giving the model a few instructions or some examples, you "programme" the model when you design your prompt. This stands apart from the majority of other NLP services, which are built for specific tasks like sentiment categorization or named entity identification. Instead, practically any task, including the creation of content or code, summarization, expansion, conversation, creative writing, style transfer, and more, can be accomplished using the completions and chat completions endpoints.

Tokens:

By dividing text into tokens, our models can comprehend and interpret it. Words or single character groups can be used as tokens. For instance, the word "hamburger" is divided into the tokens "ham", "bur", and "ger", although a short and often used word like "pear" is only one token. Numerous tokens begin with a whitespace, such as the words "hello" and "bye."

The length of both your inputs and outputs determines how many tokens are handled in a particular API request. For English text, a token roughly corresponds to 4 characters, or 0.75 words. Keep in mind that the combined length of your text prompt and produced completion cannot exceed the model's maximum context length, which is typically 2048 tokens (about 1500 words). To understand more about the conversion of text to tokens, check out our tokenizer tool.

Models:

A variety of devices with various features and pricing points power the API. Our most recent and potent model is GPT-4. The model that powers ChatGPT is GPT-3.5-Turbo is designed with conversational forms in mind. Visit our model documentation to find out more about these models and what else we have to offer.

Platform Improvements:

- Answers Endpoints
- Classification Endpoints
- Enhanced Search Endpoints
- Safety
- Prompt Library

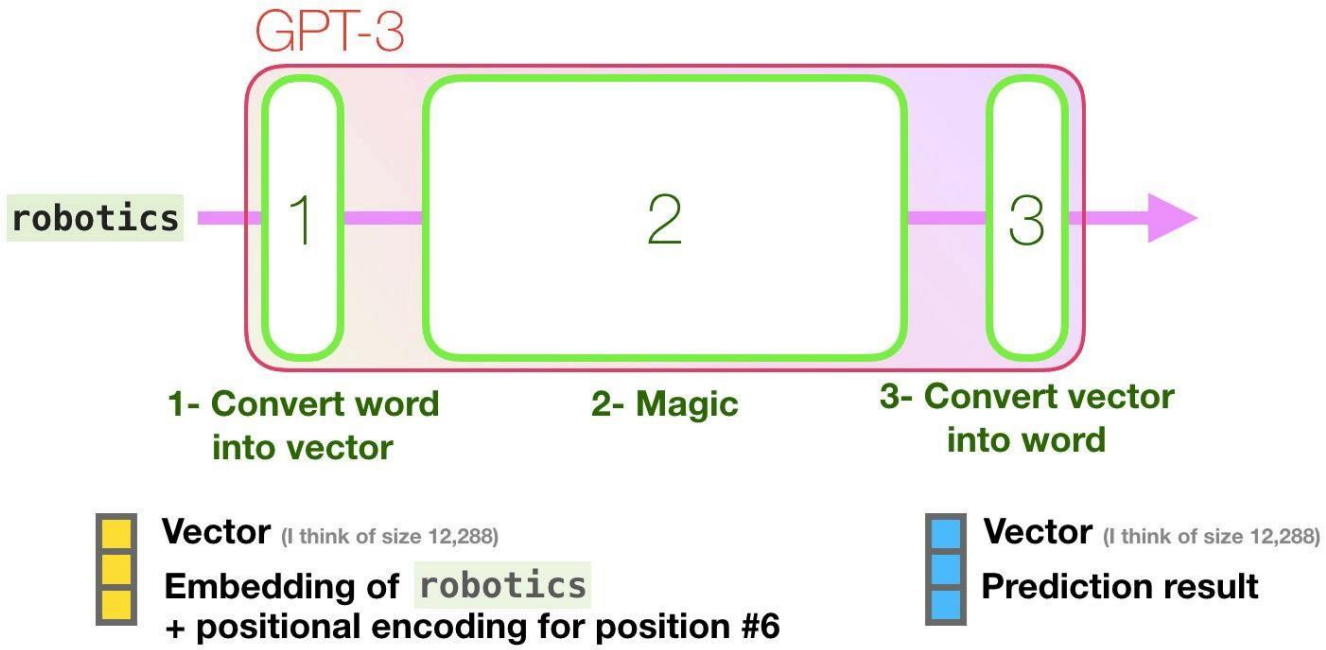

FIG 5.3.2 GPT-3 WORKING MODEL

API (Application Program Interface):

Any language can use our official Python bindings, our official Node.js library, or a community-maintained library to communicate with the API through HTTP requests

Run the following command in the directory of your Node.js project to install the official Node.js library: **install openai using npm**

Authentication:

API keys are used by the OpenAI API to authenticate users. To obtain the API key you'll need for your queries, go to your API Keys page

Keep in mind that your API key is a private key! Do not divulge it to anybody or use it in any client-side (browsers, applications) programming. Your own backend server, where your API key may be safely supplied from an environment variable or key management service, must handle production requests.

Your API key should be included in an Authorization HTTP header in every API request, as follows: **GPT_API_KEY, bearer of authorization**

Organisation requesting:

You can pass a header to an API request to specify which organisation is utilised for users who are members of several organisations. These API requests' usage will be deducted from the subscription quota for the specified organisation.

a curl example command

"GPT-Organization: YOUR_ORG_ID" and "Authorization: Bearer $OPENAI_API_KEY" are specified in the curl command at https://api..com/v1/models.

Example with the GPT Node.js package:

```
import { Configuration, GptApi } from "GPT";
const configuration = new Configuration({
  organization: "YOUR_ORG_ID",
  apiKey: process.env.GPT_API_KEY,
});
const Gpt = new GptApi(configuration);
const response = await Gpt.listEngines();
```

Example Response:


```
{
        "id":"chatcmpl-abc123",
        "object":"chat.completion",
        "created":1677858242,
        "model":"gpt-3.5-turbo-0301",
        "usage":{
          "prompt_tokens":13,
          "completion_tokens":7,
          "total_tokens":20
 },
  "choices":[
          {
            "message":{
              "role":"assistant",
              "content":"\n\nThis is a test!"
            },
            "finish_reason":"stop",
            "index":0
          }
        ]
}
```


List Models:


```
{
        "data": [
          {
                "id": "model-id-0",
                "object": "model",
                "owned_by": "organization-owner",
```

"permission": [...]
},
{
"id": "model-id-1",
"object": "model",
"owned_by": "organization-owner",
"permission": [...]
},
{
"id": "model-id-2",
"object": "model",
"owned_by": "gpt",
"permission": [...]
},
],
"object": "list"
}

Request:

Model string for the request body

Required

ID of the appropriate model. With this endpoint, you can apply either the text-davinci-edit-001 or the code-davinci-edit-001 model.

input string Optional Defaults to "

The input text that will serve as the edit's starting point. instruction string

the directive directing the model on how to modify the prompt.

n integer

Alternative Defaults to 1

How many input and instruction edits should be produced.

temperature number

Alternative Defaults to 1

Which temperature between 0 and 2 should be used for sampling? Higher numbers, such as 0.8, will result in a more random output, whilst lower values, such as 0.2, would result in a more concentrated and predictable output.

In general, we advise against changing both this and top_p.

number top_p Optional; defaults to 1

ADVANTAGES

- The most obvious advantage is that it can generate enormous amounts of text, which makes it simpler and more effective for us to create text-based content.
- It may also be used to compose essays, summarise texts, provide answers to queries, and more.
- Sales teams may use GPT-3 to reach out to new customers, customer service centres can use it to respond to customer inquiries or support chatbots, and marketing organisations can use it to create content.
- Whenever a large amount of text needs to be generated from a machine based on some small amount of text input, GPT-3 provides a good solution.

DISADVANTAGES

- It makes an effort to avoid controversial topics, yet problems can still arise.

Require more storage to safe Dataset

CHAPTER 6

CONCLUSION

6.1 CONCLUSION

We described applicable AI-based algorithms for AI Powered Recaster. We created new website when there is no openly obtainable website for Recaster. What is more, we experimented with the AI-based pre-trained Turnitin algorithm in Neural Network and also include the GTP-3. Those models followed by supervised fine-tuning and transfer learning with regularization technique to fit our specific tasks. We observed that training with a website is not sufficient to get the best accuracy based on AI approach. In addition to this, it was not enough just using regularization technique in our system. After analysing our website, we find out that the results of using AI learning and regularization can work better than those of Algorithm. After that, the performances of website are better. All of the above, our considered this website for working Professionals as well as for the student for reducing their time.

7. APPENDIX

7.1 SCREENSHOT

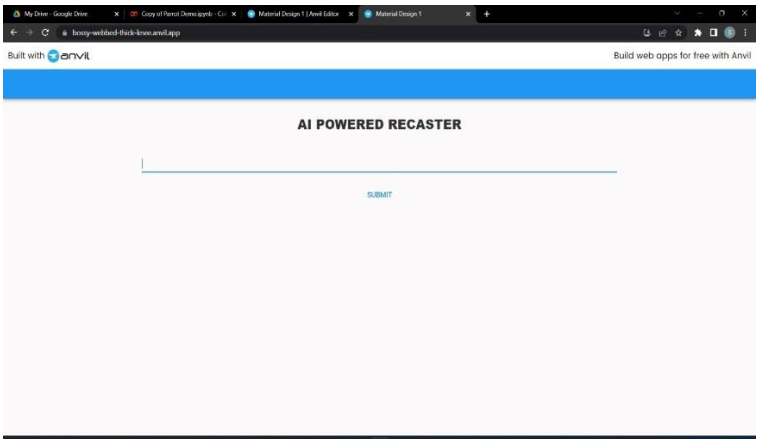

FIG 6.1.1: HOME PAGE

Fig 7.1 Shows the home page for recaster. This page contains input box as well as text area for users to provide inputs.

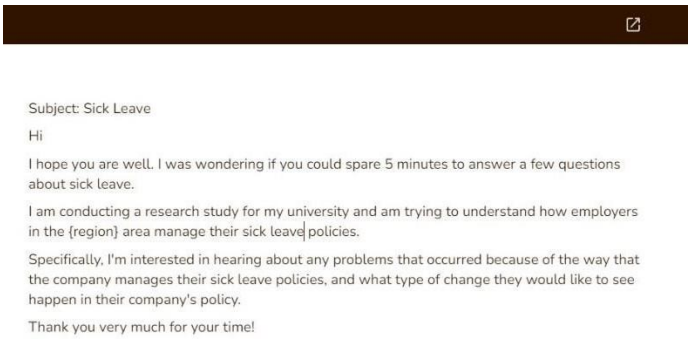

FIG 6.1.2 RECASTED RESULT

Fig 7.1.2 Shows the paraphrase output, the content matches with the key word or input given by the user.

7.2 SOURCE CODE

Index.html

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charSet="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <meta name="theme-color" content="#f4db4a" />
  <meta name="msapplication-TileColor" content="#f4db4a" />
  <meta http-equiv="X-UA-Compatible" content="IE=edge" />
  <link rel="mask-icon" href="https://storage.googleapis.com/Recaster-me/public/favicon/safaripinned-tab.svg" color="#f4db4a" />
  <link rel="apple-touch-icon" sizes="180x180" href="https://storage.googleapis.com/Recasterme/public/favicon/apple-touch-icon.png" />
  <link rel="icon" type="image/png" sizes="32x32" href="https://storage.googleapis.com/Recasterme/public/favicon/favicon-32x32.png" />
  <link rel="icon" type="image/png" sizes="16x16" href="https://storage.googleapis.com/Recasterme/public/favicon/favicon-16x16.png" />
  <link rel="shortcut icon" type="image/x-icon" href="https://storage.googleapis.com/Recasterme/public/favicon/favicon.ico" />
  <meta name="og:site_name" content="Recaster" />
  <meta name="og:type" content="website" />
  <meta name="og:image" content="https://storage.googleapis.com/Recaster-me/public/image/logospaced.png" />
  <meta name="author" content="Recaster" />
  <meta name="copyright" content="Recaster" />
 <meta name="application-name" content="Recaster" />
  <title>AI Powered Recaster &amp;amp; Writing Assistant</title>
  <meta name="og:title" content="Recaster · Best AI Writer, Content Generator &amp;amp; Writing Assistant" />
```

<meta name="description" content="Recaster is an AI writing assistant that helps you create highquality content, in just a few seconds, at a fraction of the cost!" />

<meta name="og:description" content="Recaster is an AI writing assistant that helps you create high-quality content, in just a few seconds, at a fraction of the cost!" />

<meta name="og:url" content="https://Recaster.me" />

<meta name="next-head-count" content="22" />

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />

<link rel="preload" href="/_next/static/css/c7757439036edf6f.css" as="style" />

<link rel="stylesheet" href="/_next/static/css/c7757439036edf6f.css" data-n-g="" />

<link rel="preload" href="/_next/static/css/8103cf8cfedb7d7c.css" as="style" />

<link rel="stylesheet" href="/_next/static/css/8103cf8cfedb7d7c.css" data-n-p="" /><noscript datan-css=""></noscript>

<script defer="" nomodule="" src="/_next/static/chunks/polyfills-c67a75d1b6f99dc8.js"></script>

<script src="/_next/static/chunks/webpack-ce9cc6c38c95d9d5.js" defer=""></script>

<script src="/_next/static/chunks/framework-3b5a00d5d7e8d93b.js" defer=""></script>

<script src="/_next/static/chunks/main-a8438a48ae620a80.js" defer=""></script>

<script src="/_next/static/chunks/pages/_app-170c9f0d8cf85424.js" defer=""></script>

<script src="/_next/static/chunks/664-e432d276bc67604d.js" defer=""></script>

<script src="/_next/static/chunks/757-16a8f82168621720.js" defer=""></script>

<script src="/_next/static/chunks/908-dee1f7c872932838.js" defer=""></script> <script src="/_next/static/chunks/325-75c04c4a39441cec.js" defer=""></script>

<script src="/_next/static/chunks/221-18e5c5b5312f0a7e.js" defer=""></script>

<script src="/_next/static/chunks/pages/index-c3d0f2f36da2b3b6.js" defer=""></script>

<script src="/_next/static/Nu9UwyVZ_5Pv--gKNLkh_/_buildManifest.js" defer=""></script>

<script src="/_next/static/Nu9UwyVZ_5Pv--gKNLkh_/_ssgManifest.js" defer=""></script>

<style datahref=https://fonts.googleapis.com/css2?family=Nunito:wght@400;600;700;800&display=swap >

BACKEND

```
class Adequacy():

  def __init__(self, model_tag='prithivida/parrot_adequacy_model'):
    from transformers import AutoModelForSequenceClassification, AutoTokenizer
    self.adequacy_model = AutoModelForSequenceClassification.from_pretrained(model_tag)
    self.tokenizer = AutoTokenizer.from_pretrained(model_tag)

  def filter(self, input_phrase, para_phrases, adequacy_threshold, device="cpu"):
      top_adequacy_phrases = []
      for para_phrase in para_phrases:
        x = self.tokenizer(input_phrase, para_phrase, return_tensors='pt', max_length=128, truncation=True)
        x = x.to(device)
        self.adequacy_model = self.adequacy_model.to(device)
        logits = self.adequacy_model(**x).logits
        probs = logits.softmax(dim=1)
        prob_label_is_true = probs[:,1]
        adequacy_score = prob_label_is_true.item()
        if adequacy_score >= adequacy_threshold:
            top_adequacy_phrases.append(para_phrase)
      return top_adequacy_phrases


  def score(self, input_phrase, para_phrases, adequacy_threshold, device="cpu"):
      adequacy_scores = {}
      for para_phrase in para_phrases:
        x = self.tokenizer(input_phrase, para_phrase, return_tensors='pt', max_length=128, truncation=True)
        x = x.to(device)
        self.adequacy_model = self.adequacy_model.to(device)
        logits = self.adequacy_model(**x).logits
        probs = logits.softmax(dim=1)
```

```
        prob_label_is_true = probs[:,1]
        adequacy_score = prob_label_is_true.item()
        if adequacy_score >= adequacy_threshold:
          adequacy_scores[para_phrase] = adequacy_score
      return adequacy_scores

class Fluency():
  def __init__(self, model_tag='prithivida/parrot_fluency_model'):
    from transformers import AutoModelForSequenceClassification, AutoTokenizer
    self.fluency_model = AutoModelForSequenceClassification.from_pretrained(model_tag,
num_labels=2)
    self.fluency_tokenizer = AutoTokenizer.from_pretrained(model_tag)

  def filter(self, para_phrases, fluency_threshold, device="cpu"):
      import numpy as np
      from scipy.special import softmax
      self.fluency_model = self.fluency_model.to(device)
      top_fluent_phrases = []
      for para_phrase in para_phrases:
        input_ids = self.fluency_tokenizer("Sentence: " + para_phrase, return_tensors='pt', truncation=True)
        input_ids = input_ids.to(device)
        prediction = self.fluency_model(**input_ids)
        scores = prediction[0][0].detach().cpu().numpy()
        scores = softmax(scores)
        fluency_score = scores[1] # LABEL_0 = Bad Fluency, LABEL_1 = Good Fluency
        if fluency_score >= fluency_threshold:
          top_fluent_phrases.append(para_phrase)
      return top_fluent_phrases

  def score(self, para_phrases, fluency_threshold, device="cpu"):
      import numpy as np
      from scipy.special import softmax
```

```
    self.fluency_model = self.fluency_model.to(device)
    fluency_scores = {}
    for para_phrase in para_phrases:
      input_ids = self.fluency_tokenizer("Sentence: " + para_phrase, return_tensors='pt', truncation=True)
      input_ids = input_ids.to(device)
      prediction = self.fluency_model(**input_ids)
      scores = prediction[0][0].detach().cpu().numpy()
      scores = softmax(scores)
      fluency_score = scores[1] # LABEL_0 = Bad Fluency, LABEL_1 = Good Fluency
      if fluency_score >= fluency_threshold:
        fluency_scores[para_phrase] = fluency_score
    return fluency_scores
 def rank(self, input_phrase, para_phrases, diversity_ranker='levenshtein'):
  if diversity_ranker == "levenshtein":
return self.levenshtein_ranker(input_phrase, para_phrases)
    elif diversity_ranker == "euclidean":
return self.euclidean_ranker(input_phrase, para_phrases)
    elif diversity_ranker == "diff":
   from sentence_transformers import SentenceTransformer
   self.diversity_model = SentenceTransformer(model_tag)

 def rank(self, input_phrase, para_phrases, diversity_ranker='levenshtein'):
    if diversity_ranker == "levenshtein":
      return self.levenshtein_ranker(input_phrase, para_phrases)
    elif diversity_ranker == "euclidean":
      return self.euclidean_ranker(input_phrase, para_phrases)
    elif diversity_ranker == "diff":
      return self.diff_ranker(input_phrase, para_phrases)

 def euclidean_ranker(self, input_phrase, para_phrases):
    import pandas as pd
    from sklearn_pandas import DataFrameMapper
```

```
    from sklearn.preprocessing import MinMaxScaler
    from scipy import spatial

    diversity_scores = {}
    outputs = []
    input_enc = self.diversity_model.encode(input_phrase.lower())
    for para_phrase in para_phrases:
        paraphrase_enc = self.diversity_model.encode(para_phrase.lower())
        euclidean_distance = (spatial.distance.euclidean(input_enc, paraphrase_enc))
        outputs.append((para_phrase,  euclidean_distance))
    df = pd.DataFrame(outputs, columns=['paraphrase', 'scores'])
    fields = []
    for col in df.columns:
        if col == "scores":
            tup = ([col], MinMaxScaler())
        else:
            tup = ([col], None)
        fields.append(tup)

    mapper = DataFrameMapper(fields, df_out=True)
    for index, row in mapper.fit_transform(df.copy()).iterrows():
        diversity_scores[row['paraphrase']] = row['scores']
    return  diversity_scores

def levenshtein_ranker(self, input_phrase, para_phrases):
    import Levenshtein
    diversity_scores = {}
    for para_phrase in para_phrases:
        distance = Levenshtein.distance(input_phrase.lower(), para_phrase)
        diversity_scores[para_phrase] =  distance
    return diversity_scores
```

```
def diff_ranker(self, input_phrase, para_phrases):
  import difflib
  differ = difflib.Differ()
  diversity_scores ={}
  for para_phrase in para_phrases:
     diff = differ.compare(input_phrase.split(), para_phrase.split())
     count = 0
     for d in diff:
       if "+" in d or "-" in d:
         count += 1
     diversity_scores[para_phrase] = count
```

REFERENCES

- L. Yao, N. Peng, W. Ralph, K. Knight, D. Zhao and R. Yan, **"Plan-and-write: Towards better automatic storytelling"**.
- K. Arulkumaran, A. Cully and J. Togelius, "Alphastar", **Proceedings of the Genetic and Evolutionary Computation Conference Companion,** Jul 2019**.**
- Radford, Alec, Karthik Narasimhan, Tim Salimans and Ilya Sutskever, **"Improving language understanding by generative pre-training"**, 2018.
- W. Wang, P. Li and HT. Zheng, **"Consistency and Coherency Enhanced Story Generation"**, Advances in Information Retrieval. ECIR 2021