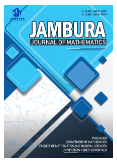
JAMBURA JOURNAL OF MATHEMATICS
Jambura J. Math. Vol. 5, No. 2, pp. 327–338, August 2023
Journal Homepage: http://ejurnal.ung.ac.id/index.php/jjom
DOI: https://doi.org/10.34312/jjom.v5i2.20574



Information Retrieval Performance in Text Generation using Knowledge from Generative Pre-trained Transformer (GPT-3)

Kaira Milani Fitria[1,*]

[1]*Faculty of Computer Science, Informatics & Business Institute Darmajaya, Lampung 35141, Indonesia*
[*]*Corresponding author. Email: kairaamilanii@gmail.com*

ABSTRACT

The rise of advanced language models like GPT-3 and text generation has witnessed remarkable progress. However, leveraging the vast amount of knowledge within these models to enhance information retrieval performance remains an area that needs to be explored. This research used Artificial Intelligence, specifically the OpenAI GPT-3 language model, to create an application to help make written content. This research investigates the impact of incorporating GPT-3's knowledge into text generation processes and evaluates its influence on information retrieval tasks. Several features in text generation generate text that requires exact information, such as specifications for a product and accurate descriptions of a job or product, which are included in the concept of information retrieval in text creation by language models. The research used the few-shot learning method in the GPT-3 language model. The generated responses are then evaluated using established information retrieval metrics such as precision, recall, and F1-score. The findings of this research reveal the effectiveness of utilizing GPT-3's knowledge in enhancing information retrieval performance. The generated responses demonstrate improved relevance to user queries, resulting in the same performance precision and recall scores compared to other paid text generator websites. Application results are testing in capabilities of retrieving some information. Application capabilities tested on other commercial text generator engines. The test results obtained BERTscore 86% (precision), 88% (recall), and 87% (F1-Score).

Keywords:
Information Retrieval; Language Model; GPT; Text Generation; Few-shot Learning

Citation Style:

K. M. Fitria, "Information Retrieval Performance in Text Generation using Knowledge from Generative Pre-trained Transformer (GPT-3)", *Jambura J. Math.*, vol. 5, No. 2, pp. 327–338, 2023, doi: https://doi.org/10.34312/jjom.v5i2.20574

1. Introduction

Artificial intelligence (AI) has become commonplace daily and rapidly expands [1]. NLP refers to the automatic computational processing of human languages as a whole. It consists of both algorithms that accept text created by humans as inputs and algorithms that generate text with a natural appearance as outputs [2]. Language models are currently helpful in various sectors of natural language processing, such as text creation, by applying the concept of "information retrieval." In the text creation category, information retirement as a concept of using the language model has a role as a program that is related to organizing, storing, retrieving, and evaluating information from

e-ISSN: 2656-1344 © 2023 K. M. Fitria | Under the license CC BY-NC 4.0
Received: 2023-06-15 | Revised: 2023-06-28 | Accepted: 2023-07-06| Published: 2023-08-01

document repositories, especially textual information. A good language model will only retrieve relevant text according to the context the user wants. An information retrieval system searches a library of natural language texts to precisely retrieve the set of documents that respond to a user's query. They started as library systems. These tools help users find the data they need, but they don't try to infer or come up with solutions. Making meaning of data requires an information retrieval system. The work of language modeling serves as the foundation for model pre-training in natural language processing (NLP). Given a past of unannotated texts, language modeling aims to predict the following token [3]. The first model to use unidirectional transformers as the framework for the GPT of language models was generative pre-training (GPT), demonstrating the striking potential of pre-training techniques for various downstream applications [4]. While there are undoubtedly many benefits, it has also created new, challenging problems that must be solved [5]. An AI model called the Generative Pre-Trained Transformer-3 (GPT-3) learned unsupervised on a significant corpus of text data. GPT-3 can now make responses in written language that mirror those produced by humans thanks to this training [6].

OpenAI created the powerful huge language model GPT-3. Through knowledge distillation, its knowledge can be applied to jobs farther down the line [7]. More substantial natural language processing (NLP) and machine learning (ML) capabilities enable more powerful models to perform with better understanding and response abilities rather than being designed for performing human communication interactions [8]. GPT-3 has demonstrated its ability to produce writing that can be mistaken for a human author, including poetry, articles, web interface code, and descriptions of products or jobs [9]. GPT-3 functions with a small amount of fine-tuning data. The language model knows practically every domain by nature. Only a few instructions on what to do and, preferably, a few instances of the desired result are used to teach it (few-shot learning) [10]. The "text-davinci-003" model now uses a training dataset of 45 million web pages, books, and other sources [11]. As previously mentioned by other authors, GPT-3 demonstrated excellent cooperation and produced pertinent, accurate, and proper research [12, 13]. The successor to GPT-2, GPT-3, increased the data scale (45 TB vs. 40 GB) and the parameter space (175 billion vs. 1.5 billion), making it the most significant language model ever made. The model performs outstandingly in zero-shot and few-shot conditions and can complete downstream tasks without fine-tuning [14]. AI models can be connected to platforms like websites via their APIs (Application Programming Interfaces) and utilized for various functions [11]. The ability to annotate data for machine learning model training is one-way GPT-3 may assist people [15]. Prompt-Learning, sometimes called Prompting, provides insight into what natural language processing (NLP) may include in the future [16].

These data repositories may contain data that text indexing and retrieval systems may index and make searchable by users. As a result, retrieval systems give users internet access to knowledge they might not be aware of, and they are not obliged to know or care about where the ability is stored. Users can use a single search to query all the data the administrator has chosen to index. The two primary operations in retrieval systems are indexing and matching. Selecting terms to represent a text is the process of indexing. The indexing process includes stemming, frequent word removal, and string tokenization. The matching process is the method for determining how similar two text representations are to one another. Making applications using AI capabilities as a provider of sentence ideas can make the work of making text easier. In this research, we created a text generator app that produces exciting and varied sentences so that users can adjust according to their needs. This application uses a language model developed by OpenAI, GPT-3, as the latest 3rd generation of Generative Pre-trained Transformer, an autoregressive language model with deep learning to produce human-like text. The features in this app that implement the information retrieval concept are product descriptions, job descriptions, email templates, and advertisements.

2. Methods

2.1. Research Framework

The frameworks in the research approach must be used in a specific order. This framework's chronological order illustrates the steps that must be followed to guarantee the success of this research. The structure is shown in Figure 1.

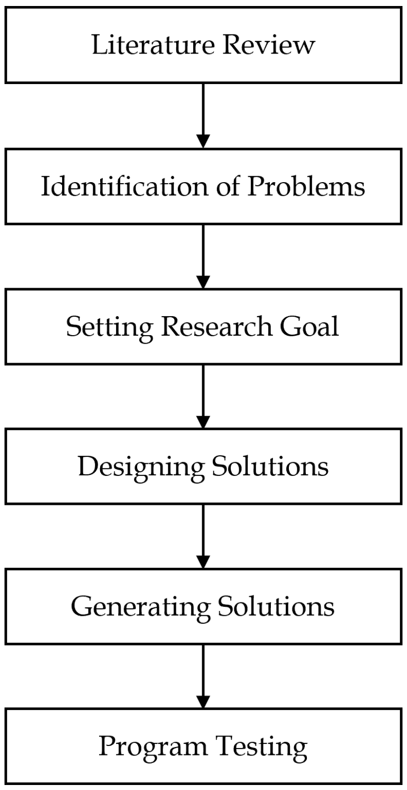


Figure 1. Research Framework

Based on Figure 1, the explanation for each step in the research framework can be described as follows:

1. Literature Review
   A literature review is a critical and systematic examination of previously published studies and academic works pertinent to a given subject or research question. Establishing the present level of knowledge on an issue, spotting gaps, disputes, or inconsistencies in the available literature, and deciding whether additional study is necessary are the main goals of a literature review. Some of the principal sources that are cited in this study include Vaswani [17] in the study ”Attention is All You Need, Advances in neural information processing systems”, Devlin [18] in the survey ”BERT: Pre-training of deep Bidirectional Transformers

for language understanding," Radford [19] in "Language Models are Unsupervised Multitask Learners," and Brown [20] in "Language models are few-shot learners."

2. Identification of Problems
   The growth of e-commerce sites raises a concern about competitiveness in the marketing process, where the study's identification of difficulties begins. Copywriting abilities are necessary for a digital business. Making an application that can offer creative sentence suggestions and utilize Artificial Intelligence (AI) to make it simpler for copywriters to generate sentences is the answer to improving copywriting talents for regular people.
3. Setting Research Goal
   Based on the problem identification findings and the research's driving force, the emphasis was chosen. The development of a publicly accessible text generator application is the primary goal of this study. The application design also includes some auxiliary elements, including descriptions of goods and services, job descriptions, social network captions, email marketing templates, adverts, and suggestions for business promotion. To increase promotional content insight, features are also built to produce compelling reports for promotional videos without negating the need for more enticing marketing strategies, YouTube video title suggestions, word and sentence improvements, paragraph summaries, and keyword generation from a description.
4. Designing Solutions
   The copywriting application development approach is used to undertake system design and application development while designing solutions based on the research topic. By practicing with the language model and utilizing the API code or service from this language model in the previously created web program code, the GPT-3 language model can be implemented in the application.
5. Generating Simulations
   Designing the virtual world, developing and implementing the appropriate algorithms and models, incorporating user interactions, and ensuring the simulation behaves realistically or as intended are all steps in generating simulations for an app. Expertise in user interface design, computer graphics, physics modeling, and artificial intelligence is needed. Overall, creating simulations for an app offers a dynamic and interactive element that enables users to interact with digital representations of real-world situations, scenarios, or systems. Depending on the precise goals and objectives of the app, it improves user engagement, helps to learn, allows for experimentation, and offers useful functionality.
6. Program Testing
   Program testing entails thoroughly examining an application to find flaws, mistakes, or problems and ensuring it works as intended. Program testing's primary goal is to increase usability, dependability, and quality before an app is made available to customers. The objectives, scope, scenarios, cases, and resources needed for testing are all described in this plan. It ensures that testing procedures are well-planned and covers all important facets of the application. User acceptance testing entails testing the app with stakeholders or end users to ensure it satisfies their needs and expectations. To imitate natural usage conditions, it frequently occurs in a real-world setting.

2.1.1. Application Design

The main objective of this system is to create a web application that acts as a bridge between the language model API service and the user via the User Interface (UI). The website interface will collect user input, utilize this information as a prompt for the GPT-3 language model service, and subsequently present the results generated by the language model on the user's visible web interface.

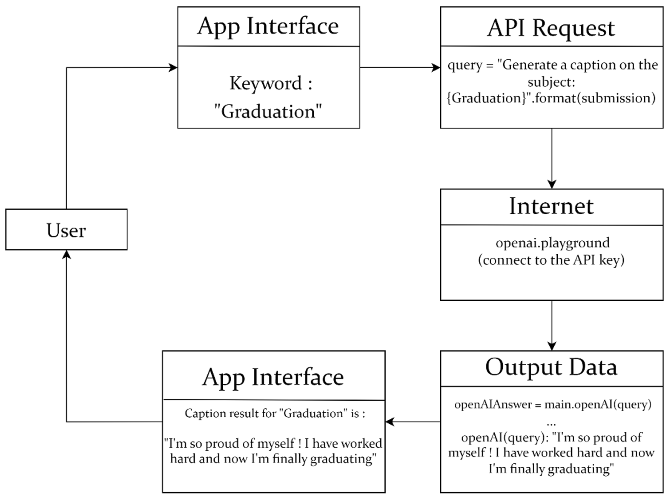

App Interface
Keyword : "Graduation"
API Request
query = "Generate a caption on the subject: {Graduation}".format(submission)
Internet
openai.playground (connect to the API key)
User
Output Data
openAIAnswer = main.openAI(query)
...
openAI(query): "I'm so proud of myself ! I have worked hard and now I'm finally graduating"
App Interface
Caption result for "Graduation" is :
"I'm so proud of myself ! I have worked hard and now I'm finally graduating"

Figure 2. System Overview

The application development process involves creating app prototypes, demonstrating app functionalities, and conducting app testing. The application's design concept is implemented by training the language model and incorporating API code or services from the language model into the existing web program code. The primary task is to establish a connection between the web app, the language model API service, and the user through the User Interface (UI). The website interface will collect user input, utilize it as a prompt for the GPT-3 language model service, and then display the results generated by the language model back to the user on the visible web interface.

The GPT-3 training in this app uses the few-shot learning method. The few-shot learning method starts by giving a prompt containing commands. Then the results are directed as desired. Model specifications used for the training process using the Complete Mode, text-davinci-002 model, the Temperature is 0.5 with a Maximum length of 256, and the Top-P is 1.

The whole application is scripted in Visual Studio Code software. The program code is Python 3.8, using several supporting libraries such as Flask and OpenAI. For web display (frontend) using HTML, CSS, and JavaScript programming languages. In designing the website layout, the Adobe Illustrator application created detailed graphic components and prepared a prototype of a website layout using the Figma application. Application testing before deployment runs on localhost by running the script program in the Command Prompt or Visual Studio Code which is open in the Chrome browser. After the application runs well on localhost, the program folder is deployed to the PythonAnywhere website so that the website has a URL that various devices can access. Android application built in Android Studio by converting the website layout to a mobile form. Android-based application releases are done through the author's developer account on the Google Play store platform.

2.2. Application Testing

The testing phase of this application consists of three tests, which are evaluated using an evaluation matrix. The researcher selects the most suitable evaluation matrix for the language model, BERTscore. The evaluation results are calculated by considering the features and gathering test data for each feature to assess the performance. Each feature is then categorized and tested using BERTscore, particularly when measuring the similarity score of ”context” or sentence meaning. Figure 3 illustrates the testing process using BERTscore evaluation matrices for the created applications.

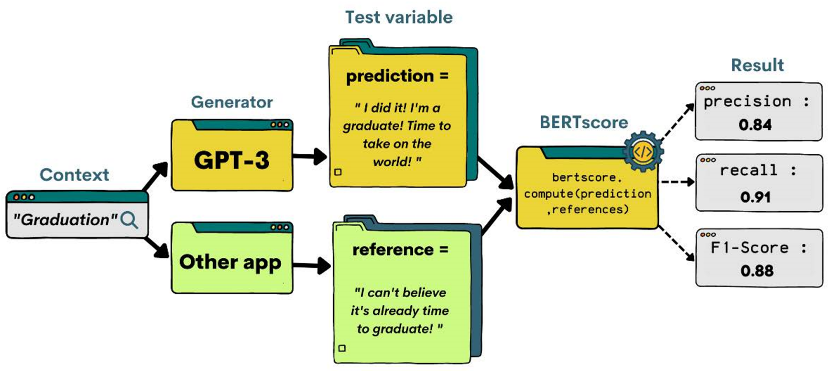

Test variable
Result
prediction =
" I did it! I'm a graduate! Time to take on the world! "
Generator
GPT-3
BERTscore
precision : 0.84
recall : 0.91
F1-Score : 0.88
bertscore.compute(prediction,references)
Context
"Graduation"
Other app
reference =
"I can't believe it's already time to graduate! "

Figure 3. BERTscore test block diagram

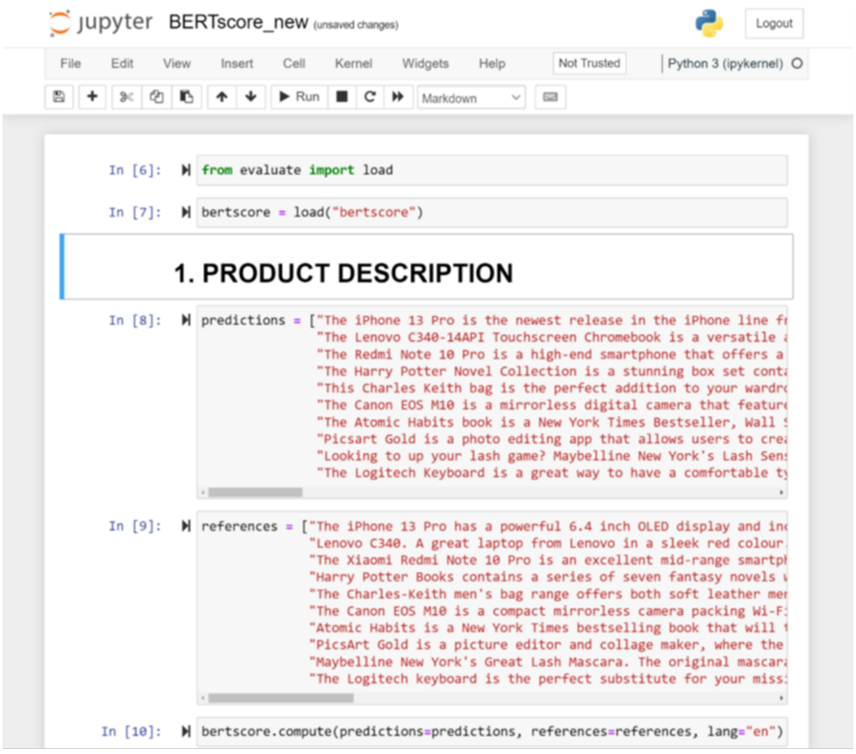
Two types of data are tested: ”predictions” data obtained from sentences generated by the text generator app and ”references” data obtained from sentences generated by other commercial applications. Once the data for testing is acquired, the tests are conducted with specific parameters for each feature. The BERTscore python libraries are employed in the testing process, executed through Jupyter Notebook. A program is committed to displaying the results of the text generator application’s sentence quality testing and determining whether the quality is comparable to that of other text generator applications.

3. Results and Discussions

The application test results obtained BERTscore scores for each of the features being compared, like the Product Description, Job Description, Email Template, and Advertisement features. BERTscore testing conducted the precision, recall, and F1 score results. All of them are evaluation metrics commonly used in information retrieval and classification tasks. They provide insights into the performance and effectiveness of a model or system. Precision is a metric that measures the overall correctness of predictions or classifications. Precision offers a general understanding of how well the model performs across all tasks. In information retrieval tasks, recall calculates the proportion of relevant documents retrieved from all relevant documents in the dataset. Recall indicates how effectively the system retrieves all relevant documents in response to a user query. High recall suggests that the system is effective in finding relevant results. The F1 score is a metric that combines precision and recall into a single value, providing a balanced measure of a model’s performance. The F1 score is helpful when there is an imbalance between precision and recall. F1-Score delivers a comprehensive evaluation of a model’s effectiveness by considering both the ability to retrieve relevant instances (recall) and the accuracy of those retrievals (precision). It is often used as a primary metric when optimizing models for information retrieval tasks.

3.1. Product Description

In product description testing, the text data successfully generated by the application compared its performance with the generated text by another text generator application (copy.ai). The text generated by the app is in the ”predictions” variable, and the text generated by the web copy.ai is in the ”references” variable. One of the tests is using the prompt text “iphone 13 pro” and the results is tested in Figure 4.


jupyter BERTscore_new (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown

```
In [6]:  from evaluate import load

In [7]:  bertscore = load("bertscore")
```

1. PRODUCT DESCRIPTION

```
In [8]:  predictions = ["The iPhone 13 Pro is the newest release in the iPhone line f
                        "The Lenovo C340-14API Touchscreen Chromebook is a versatile
                        "The Redmi Note 10 Pro is a high-end smartphone that offers a
                        "The Harry Potter Novel Collection is a stunning box set cont
                        "This Charles Keith bag is the perfect addition to your wardr
                        "The Canon EOS M10 is a mirrorless digital camera that featur
                        "The Atomic Habits book is a New York Times Bestseller, Wall
                        "Picsart Gold is a photo editing app that allows users to cre
                        "Looking to up your lash game? Maybelline New York's Lash Sen
                        "The Logitech Keyboard is a great way to have a comfortable t

In [9]:  references = ["The iPhone 13 Pro has a powerful 6.4 inch OLED display and in
                       "Lenovo C340. A great laptop from Lenovo in a sleek red colour
                       "The Xiaomi Redmi Note 10 Pro is an excellent mid-range smartp
                       "Harry Potter Books contains a series of seven fantasy novels
                       "The Charles-Keith men's bag range offers both soft leather me
                       "The Canon EOS M10 is a compact mirrorless camera packing Wi-F
                       "Atomic Habits is a New York Times bestselling book that will
                       "PicsArt Gold is a picture editor and collage maker, where the
                       "Maybelline New York's Great Lash Mascara. The original mascar
                       "The Logitech keyboard is the perfect substitute for your miss

In [10]: bertscore.compute(predictions=predictions, references=references, lang="en")
```

Figure 4. BERTscore calculation on the Product Description feature

The test was conducted by collecting the result text ten times with different text. The BERTscore calculation results in the Product Description feature is the Precision, Recall, and F1-score, which can be seen in Table 1.

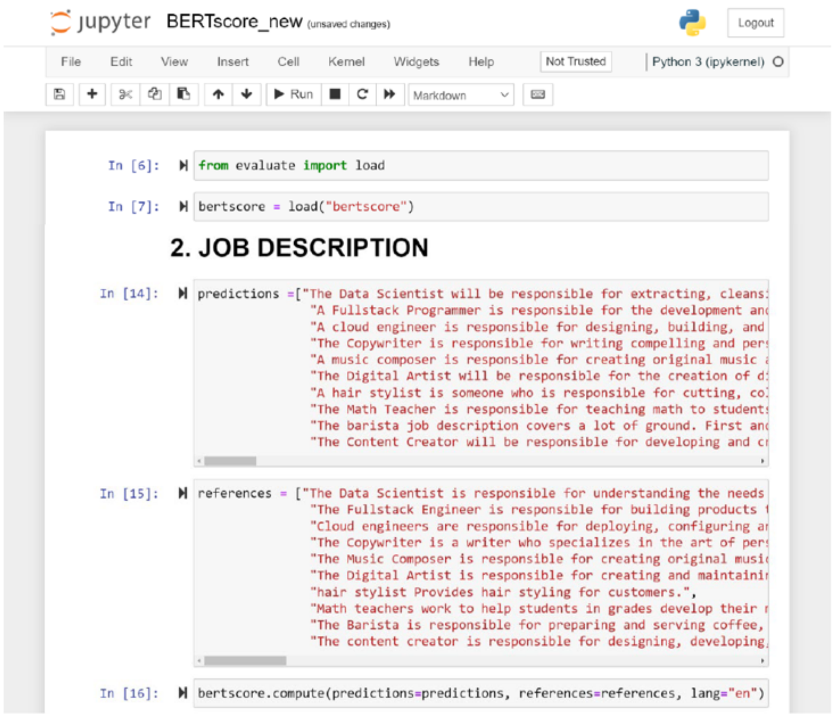
Table 1. BERTscore result from Product Description feature

Precision	Recall	F1-Score
0,84986192	0,91306293	0,880329549
0,857682824	0,892686009	0,874834418
0,879660964	0,909389794	0,894278347
0,903172612	0,912778378	0,907950103
0,848058701	0,854429066	0,851231992
0,876385868	0,928487837	0,901684821
0,878456295	0,857518733	0,867861271
0,885455251	0,896008193	0,890700459
0,868916512	0,915061235	0,891392112
0,881982982	0,899940133	0,890871108
$\overline{x}_{pr}$= 0,872963393	$\overline{x}_{re}$= 0,897936231	$\overline{x}_{F1}$= 0,885113418

The BERTscore results on the Product Description feature, namely the Precision, Recall, and F1-score values, are (0.87); (0.89); (0.88). The maximum weight for these three evaluation matrices is (1.00), so the performance of the application in creating product descriptions has performance equivalent to the text generator copy.ai with a precision percentage of 87%, recall of 89%, and F1 score of 88 %.

3.2. *Job Description*

In job description testing, the text data successfully generated by the application compared its performance with the generated text by another text generator application (rytr.me). The text generated by the app is in the ”predictions” variable, and the text generated by the web rytr.me is in the ”references” variable. One of the tests is using the prompt text “data scientist” and the results is tested in Figure 5.



```
In [6]:  from evaluate import load

In [7]:  bertscore = load("bertscore")
```

2. JOB DESCRIPTION

```
In [14]: predictions =["The Data Scientist will be responsible for extracting, cleans
                      "A Fullstack Programmer is responsible for the development an
                      "A cloud engineer is responsible for designing, building, and
                      "The Copywriter is responsible for writing compelling and per
                      "A music composer is responsible for creating original music
                      "The Digital Artist will be responsible for the creation of d
                      "A hair stylist is someone who is responsible for cutting, co
                      "The Math Teacher is responsible for teaching math to student
                      "The barista job description covers a lot of ground. First an
                      "The Content Creator will be responsible for developing and c

In [15]: references = ["The Data Scientist is responsible for understanding the needs
                      "The Fullstack Engineer is responsible for building products
                      "Cloud engineers are responsible for deploying, configuring a
                      "The Copywriter is a writer who specializes in the art of per
                      "The Music Composer is responsible for creating original musi
                      "The Digital Artist is responsible for creating and maintaini
                      "hair stylist Provides hair styling for customers.",
                      "Math teachers work to help students in grades develop their
                      "The Barista is responsible for preparing and serving coffee,
                      "The content creator is responsible for designing, developing

In [16]: bertscore.compute(predictions=predictions, references=references, lang="en")
```

Figure 5. BERTscore calculation on the Job Description feature

The test was conducted by collecting the result text ten times with different text. The BERTscore calculation results in the Job Description feature is the Precision, Recall, and F1-score, which can be seen in Table 2.

The BERTscore results for the Job Description feature, namely the Precision, Recall, and F1 scores, are (0.86); (0.89); (0.88). The maximum score for these three evaluation matrices is (1.00), so the performance of the application in creating professional descriptions is almost equivalent to the version of the Rytr.me text generator with a percentage of precision of 86%, recall of 89%, and F1 score of 88%.

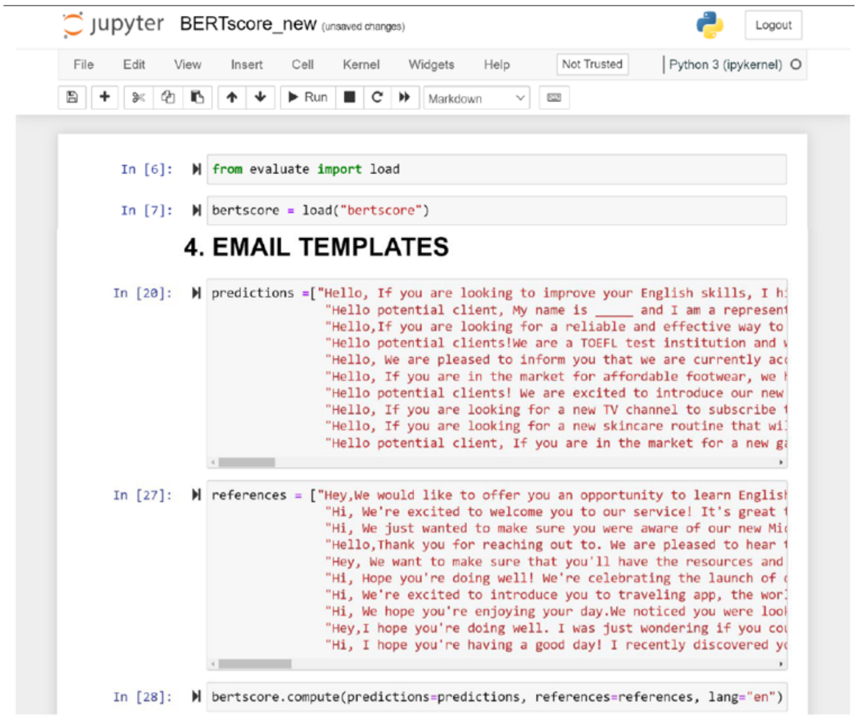
3.2.1. Email Template

In testing the email template feature, the text data generated by the application is compared with the performance of text generated by another text generator application,

Table 2. BERTscore result from Job Description feature

Precision	Recall	F1-Score
0,862736285	0,892322361	0,877279937
0,836116135	0,873488665	0,854393959
0,883280337	0,916931391	0,89979142
0,89290452	0,908216417	0,90049535
0,881690025	0,891117334	0,886378646
0,869900703	0,881533265	0,87567836
0,831672013	0,877690196	0,854061663
0,889769375	0,903478146	0,896571398
0,874397457	0,923839271	0,898438692
0,874739587	0,926254988	0,899760485
$\overline{x}_{pr}$= 0,869720644	$\overline{x}_{re}$= 0,899487203	$\overline{x}_{F1}$= 0,884284991

namely Rytr.me. The text generated by the application created is included in the ”predictions” variable, and the text generated by the Rytr.me web is included in the ”references” variable. One of the tests was carried out by entering the keyword ”cloud service” and both results were tested in Figure 6.



```
In [6]:  from evaluate import load

In [7]:  bertscore = load("bertscore")
```

4. EMAIL TEMPLATES

```
In [20]: predictions =["Hello, If you are looking to improve your English skills, I h
                       "Hello potential client, My name is _____ and I am a represent
                       "Hello,If you are looking for a reliable and effective way to
                       "Hello potential clients!We are a TOEFL test institution and
                       "Hello, We are pleased to inform you that we are currently ac
                       "Hello, If you are in the market for affordable footwear, we
                       "Hello potential clients! We are excited to introduce our new
                       "Hello, If you are looking for a new TV channel to subscribe
                       "Hello, If you are looking for a new skincare routine that wi
                       "Hello potential client, If you are in the market for a new g

In [27]: references = ["Hey,We would like to offer you an opportunity to learn Englis
                       "Hi, We're excited to welcome you to our service! It's great
                       "Hi, We just wanted to make sure you were aware of our new Mi
                       "Hello,Thank you for reaching out to. We are pleased to hear
                       "Hey, We want to make sure that you'll have the resources and
                       "Hi, Hope you're doing well! We're celebrating the launch of
                       "Hi, We're excited to introduce you to traveling app, the wor
                       "Hi, We hope you're enjoying your day.We noticed you were loo
                       "Hey,I hope you're doing well. I was just wondering if you co
                       "Hi, I hope you're having a good day! I recently discovered y

In [28]: bertscore.compute(predictions=predictions, references=references, lang="en")
```

Figure 6. BERTscore calculation on the Email Templates feature

The test was conducted by collecting the result text ten times with different text. The BERTscore calculation results in the Email Template feature is the Precision, Recall, and F1-score, which can be seen in Table 3.

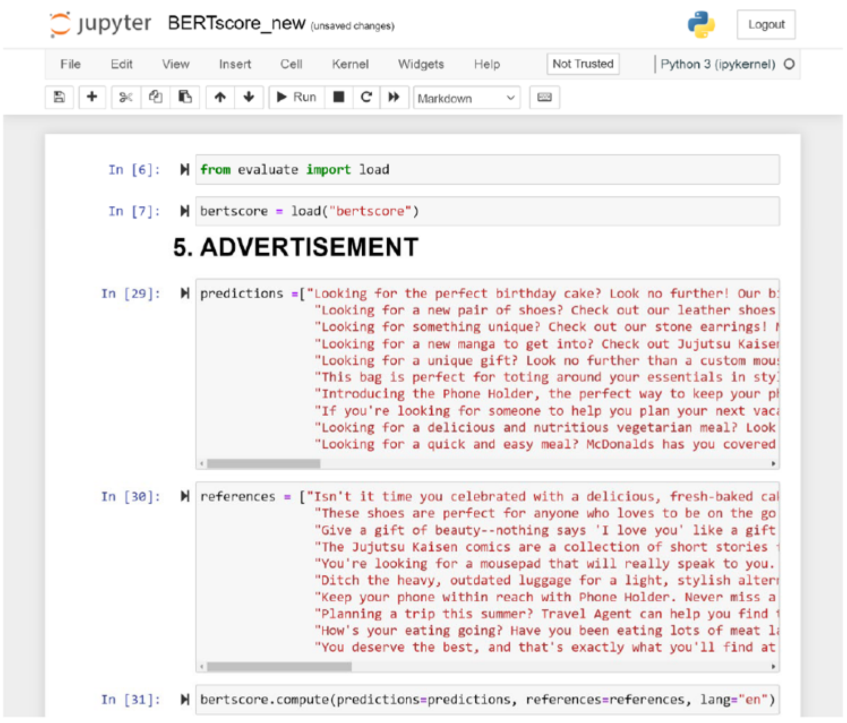
The BERTscore results for the Email Template feature, namely the Precision, Recall, and F1 scores, are (0.85); (0.85); (0.85). The maximum score for these three evaluation matrices is (1.00), so the performance of the application in creating email templates is almost equivalent to the performance of the Rytr.me text generator with a percentage of precision of 85%, recall of 85%, and F1 score of 85%.

Table 3. BERTscore result from Email Template feature

Precision	Recall	F1-Score
0,862865984	0,858090997	0,860471845
0,837025166	0,844651222	0,840820909
0,861696064	0,872532964	0,867080629
0,875247836	0,861192048	0,86816299
0,871103764	0,857683361	0,864341497
0,843739152	0,854369342	0,849021018
0,864192903	0,869520903	0,866848707
0,863211572	0,835488021	0,849123538
0,862431824	0,859217286	0,860821545
0,8444525	0,841072559	0,842759192
$\overline{x}_{pr}$= 0,858596677	$\overline{x}_{re}$= 0,85538187	$\overline{x}_{F1}$= 0,856945187

3.2.2. Advertisement

In testing the advertisement feature, the text data generated by the application is compared with the performance of text generated by another text generator application, namely Rytr.me. The text generated by the application created is included in the "predictions" variable, and the text generated by rytr.me web is included in the "references" variable. One of the tests was carried out by entering the keyword "birthday cake" with the testing in Figure 7.



```
In [6]:  from evaluate import load

In [7]:  bertscore = load("bertscore")
```

5. ADVERTISEMENT

```
In [29]:  predictions =["Looking for the perfect birthday cake? Look no further! Our b
                        "Looking for a new pair of shoes? Check out our leather shoes
                        "Looking for something unique? Check out our stone earrings! 
                        "Looking for a new manga to get into? Check out Jujutsu Kaise
                        "Looking for a unique gift? Look no further than a custom mou
                        "This bag is perfect for toting around your essentials in sty
                        "Introducing the Phone Holder, the perfect way to keep your p
                        "If you're looking for someone to help you plan your next vac
                        "Looking for a delicious and nutritious vegetarian meal? Look
                        "Looking for a quick and easy meal? McDonalds has you covered

In [30]:  references = ["Isn't it time you celebrated with a delicious, fresh-baked ca
                        "These shoes are perfect for anyone who loves to be on the go
                        "Give a gift of beauty--nothing says 'I love you' like a gift
                        "The Jujutsu Kaisen comics are a collection of short stories 
                        "You're looking for a mousepad that will really speak to you.
                        "Ditch the heavy, outdated luggage for a light, stylish alter
                        "Keep your phone within reach with Phone Holder. Never miss a
                        "Planning a trip this summer? Travel Agent can help you find 
                        "How's your eating going? Have you been eating lots of meat l
                        "You deserve the best, and that's exactly what you'll find at

In [31]:  bertscore.compute(predictions=predictions, references=references, lang="en")
```

Figure 7. BERTscore calculation on the Advertisement feature

The test was conducted by collecting the result text ten times with different text. The BERTscore calculation results in the Adventisement feature is the Precision, Recall, and F1-score, which can be seen in Table 4.

The BERTscore results for the Advertisement feature, namely the Precision, Recall, and

Table 4. BERTscore result from Advertisement feature

Precision	Recall	F1-Score
0,846525073	0,858430743	0,852436304
0,89097923	0,884832203	0,887895048
0,878361583	0,850004494	0,863950372
0,872308552	0,85904485	0,865625858
0,90151757	0,913936198	0,907684445
0,858431697	0,870725751	0,864535034
0,87516439	0,907044828	0,89081949
0,878473282	0,901118398	0,889651775
0,879751563	0,890019596	0,884855807
0,887426496	0,888258338	0,887842238
$\bar{x}_{pr}$= 0,876893944	$\bar{x}_{re}$= 0,88234154	$\bar{x}_{F1}$= 0,879529637

F1 scores, are (0.85); (0.85); (0.85). The maximum score for these three evaluation matrices is (1.00), so the performance of the application in creating advertisement is almost equivalent to the performance of the Rytr.me text generator with a percentage of precision of 85%, recall of 85%, and F1 score of 85%.

The overall score of the matrix evaluation feature testing of results can be seen in Table 5 for the BERTscore result.

Table 5. Total BERTscore result

Feature	Precision	Recall	F1-Score
Product Description	0,872963393	0,897936231	0,885113418
Job Description	0,869720644	0,899487203	0,884284991
Email Template	0,858596677	0,85538187	0,856945187
Advertisement	0,876893944	0,88234154	0,879529637
	$\bar{x}_{pr}$= 0,869543665	$\bar{x}_{re}$= 0,883786711	$\bar{x}_{F1}$= 0,876468308

The BERTscore measurement results on the feature give an average of 86% for precision, 88% for recall, and 87% for F1-scores.

4. Conclusion

Tests on the Product Description, Job Description, Email Template and Advertisement features in the application give an average BERTscore and the accuracy results provide an overall measure of correctness which give a score of 86%, and recall results assess the ability to retrieve relevant instances of the text generator model with a score of 88%. Also, the F1 score combines precision and recalls to provide a balanced evaluation and gives the final score result of 87%. These metrics assess and compare the performance of models, systems, or algorithms in information retrieval and classification tasks, allowing for informed decision-making and optimization. All of these scores are good for the text generator category and give the same performance as other commercial text generators that have been tested before.

References

[1] Z. Latinovic and S. C. Chatterjee, “Achieving the promise of ai and ml in delivering economic and relational customer value in b2b,” *Journal of Business Research*, vol. 144, pp. 966–974, 2022, doi: 10.1016/j.jbusres.2022.01.052.

[2] M. Bahja, "Natural language processing applications in business," 2021, doi: 10.5772/intechopen.92203.

[3] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech & Language*, vol. 13, no. 4, pp. 359–393, 1999, doi: 10.1006/csla.1999.0128.

[4] A. Radford, K. Narashiman, T. Salimans, and I. Sutskever1, "Improving language understanding by generative pre-training," *OpenAI*, 2018, [online] available: https://openai.com/research/language-unsupervised.

[5] C. M. Gevaert, M. Carman, B. Rosman, Y. Georgiadou, and R. Soden, "Fairness and accountability of ai in disaster risk management: Opportunities and challenges," *Patterns*, vol. 2, no. 11, p. 100363, 2021, doi: 10.1016/j.patter.2021.100363.

[6] A. Chan, "Gpt-3 and instructgpt: technological dystopianism, utopianism, and "contextual" perspectives in ai ethics and industry," *AI and Ethics*, vol. 3, no. 1, pp. 53–64, 2023, doi: 10.1007/s43681-022-00148-6.

[7] S. Y. Kim, H. Park, K. Shin, and K. Kim, "Ask me what you need: Product retrieval using knowledge from gpt-3," *arxiv*, 2022, [online] available: http://arxiv.org/abs/2207.02516.

[8] G. P. Transformer, A. O. Thunström, and S. Steingrimsson, "Can gpt-3 write an academic paper on itself, with minimal human input?" *HAL open science*, vol. 1, p. 03701250, 2022.

[9] R. Dale, "Gpt-3: What's it good for?" *Natural Language Engineering*, vol. 27, no. 1, pp. 113–118, 2021, doi: 10.1017/S1351324920000601.

[10] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples," *ACM Computing Surveys*, vol. 53, no. 3, pp. 1–34, 2021, doi: 10.1145/3386252.

[11] D. Haluza and D. Jungwirth, "Artificial intelligence and ten societal megatrends: An exploratory study using gpt-3," *Systems*, vol. 11, no. 3, p. 120, 2023, doi: 10.3390/systems11030120.

[12] R. Singh and V. Garg, "Human factors in nde 4.0 development decisions," *Journal of Nondestructive Evaluation*, vol. 40, no. 3, p. 71, 2021, doi: 10.1007/s10921-021-00808-3.

[13] T. J. Ackermann, "Gpt-3: a robot wrote this entire article. are you scared yet, human?" *Artificial Intelligence: ANI, LogicGate Computing, AGI, ASI*, 2020.

[14] M. Zhang and J. Li, "A commentary of gpt-3 in mit technology review 2021," *Fundamental Research*, vol. 1, no. 6, pp. 831–833, 2021, doi: 10.1016/j.fmre.2021.11.011.

[15] B. Ding, C. Qin, L. Liu, L. Bing, S. Joty, and B. Li, "Is gpt-3 a good data annotator?" *arxiv*, 2022, [online] available: http://arxiv.org/abs/2212.10450.

[16] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning." Association for Computational Linguistics, 2021, pp. 3045–3059, doi: 10.18653/v1/2021.emnlp-main.243.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arxiv*, vol. 2, 2019, [online] available: https://arxiv.org/abs/1810.04805.

[19] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *paperswithcode*, 2019, [online] available: https://paperswithcode.com/paper/language-models-are-unsupervised-multitask.

[20] T. B. Brown, "Language models are few-shot learners," *Computation and Language*, vol. 2, 2020.