Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (173-184)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.54838

JURNAL TEKNIK INFORMATIKA

Homepage : http://journal.uinjkt.ac.id/index.php/ti

Evaluating GPT-2-Based Keyword Generation for Scientific Abstracts Using Jaccard and Cosine Similarity

Suyono[1*], Roslina[2], Rika Rosnelly[3]

[1,3]Computer Science, Potensi Utama University, Medan, Indonesia
[2]Computer Science, State Polytechnic of Medan, Medan, Indonesia

Article:
Accepted: March 14, 2025
Revised: February 10, 2025
Issued: April 30, 2025

© Suyono et al, (2025).

This is an open-access article under the CC BY-SA license

***Correspondence Address:**
suyonosubrata@gmail.com

ABSTRACT

Author-assigned keywords play an important role in the indexing, discovery, and visibility of scientific articles, yet they often do not fully represent the content of the abstract. This research evaluates a keyword generation pipeline that combines Generative Pre-trained Transformer 2 (GPT-2) with Named Entity Recognition (NER) and Term Frequency–Inverse Document Frequency (TF-IDF) weighting to generate keywords from scientific abstracts. The pipeline was applied to a random sample of 5,667 IEEE Xplore abstracts. The generated keywords and the original author keywords were pre-processed (cleaning, casefolding, normalization, tokenization, stopword removal, and stemming) and compared using Jaccard Similarity on keyword token sets and Cosine Similarity on TF-IDF vectors. A keyword pair was classified as matched when its similarity reached a threshold of 0.1, selected through a sensitivity analysis. At this threshold, 41.50% of keyword pairs were matched under Cosine Similarity, compared with 32.87% under Jaccard Similarity, showing that the vector-based measure is more tolerant of partial overlap than exact token matching. Most pairs, however, had low similarity scores, which is consistent with the fact that a substantial share of author keywords does not appear in the abstract itself. These findings indicate that GPT-2-based keyword generation can partly reproduce author keywords and that the choice of similarity metric strongly affects the evaluation outcome, while comparison with established keyword extractors remains necessary.

Keywords: Keyword Generation; GPT-2; Named Entity Recognition; TF-IDF; Cosine Similarity; Jaccard Similarity.

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (173-184)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.54838

1. INTRODUCTION

Natural Language Processing (NLP), a subfield of artificial intelligence, has become an essential tool for understanding and processing human language in its original form [1]. NLP's ability to extract important features from textual data through techniques such as tokenisation, stemming, lemmatisation, and named entity recognition opens up opportunities to address various text processing challenges, including in the context of scientific literature [2], [3].

In the realm of scientific literature, research article abstracts play a critical role as a concise representation of the overall research content. However, abstract effectiveness is highly dependent on the accuracy of the accompanying keywords, where inaccurate or unrepresentative keywords can limit the visibility of an article in literature searches [4]. In fact, the right keywords are key to indexing, identifying the research focus, and enabling other researchers in related fields to find the article [5], [6].

In practice, author-assigned keywords in large bibliographic databases such as IEEE Xplore often do not fully represent the content of the abstract. The lexical overlap between an abstract and its keywords is therefore low, and the article becomes harder to retrieve through keyword-based indexing and search systems [4], [5]. This low keyword precision, namely the mismatch between the concepts expressed in an abstract and the keywords attached to it, is the problem that underlies this research. Based on previous research [7], NLP-based models can address this issue by automatically extracting keywords from research abstracts [8]. This approach allows for a deeper analysis of the text to identify key concepts and select the most relevant and representative keywords.

The Generative Pre-trained Transformer (GPT) developed by OpenAI offers a more advanced approach to text processing [9], [10]. GPT models, including GPT-2, utilise a transformer architecture trained on large-scale text data, making them highly effective at extracting and generating textual features due to their ability to understand context and produce coherent output [11], [12]. Later generations such as GPT-3 have been shown to generate contextual meaning and responses in tasks such as translation and question answering [13]; however, GPT-3 is available only through a commercial API and its weights are not released. This research therefore uses GPT-2, whose pre-trained weights are openly available, so that the model can be run locally on the entire dataset and the experiment can be reproduced. The findings on GPT-3 are cited only as evidence of the capability of the GPT family, not as the model used in this study.

Previous studies have addressed keyword extraction and text similarity mainly with frequency-based and encoder-based approaches. TF-IDF and Bag-of-Words representations are still widely used for text representation and classification [14], [15], and vector-space similarity based on TF-IDF has been applied to measure the similarity of news articles [1]. Transformer-based keyword extraction has also been explored, for example with BERT [7], and keyword extractors have been evaluated against real search queries [4]. However, these studies rely either on frequency-based weighting alone or on encoder-based transformers, and few of them evaluate keywords produced by a generative transformer against author-assigned keywords using both lexical (Jaccard) and vector-based (Cosine) similarity. This research addresses that gap by combining GPT-2 with Named Entity Recognition (NER) and TF-IDF, and by evaluating the generated keywords against the original author keywords on a large collection of IEEE Xplore abstracts.

Based on this foundation, this research aims to evaluate how well keywords generated from research abstracts by a GPT-2-based pipeline match the keywords assigned by the authors. GPT-2 was chosen as the core tool in this study because pre-trained generative transformers have been shown to produce relevant titles and text content automatically [16], [17]. The generated keywords are compared with the author-assigned keywords using two approaches: Jaccard Similarity, a lexical baseline that assesses similarity based on exact keyword overlap, and Cosine Similarity on TF-IDF vectors, a vector-space approach that evaluates the proximity between keyword sets.

The contributions of this research are threefold. First, it presents a keyword generation pipeline that combines GPT-2 with NER and TF-IDF weighting to generate keywords directly from raw research abstracts. Second, it evaluates the agreement between the

generated keywords and the author-assigned keywords on 5,667 IEEE Xplore abstracts using Jaccard and Cosine Similarity with a threshold-based accuracy measure. Third, it shows that Cosine Similarity is more tolerant than Jaccard Similarity in recognizing partially matching keywords, and that the achievable similarity is limited by author keywords that do not appear in the abstract [5]. These findings provide a basis for designing and evaluating automatic keyword generation systems for scientific repositories.

2. METHODS

This research focuses on the use of the GPT-2 model as a generative transformer-based keyword extraction technique applied to research abstracts used as the research dataset. The keywords generated by the GPT-2 model are then compared with the original keywords listed in the abstract dataset to evaluate the degree of consistency and relevance between the model's predicted results and the author-defined keywords.

The method used in this study is a quantitative experimental method, in which the performance of the GPT-2 model is empirically evaluated using two similarity metrics, namely Jaccard Similarity and Cosine Similarity, to measure the accuracy level of similarity between the keywords extracted by GPT-2 and the reference keywords provided by the authors. Figure 1 displays the workflow of the research stages:

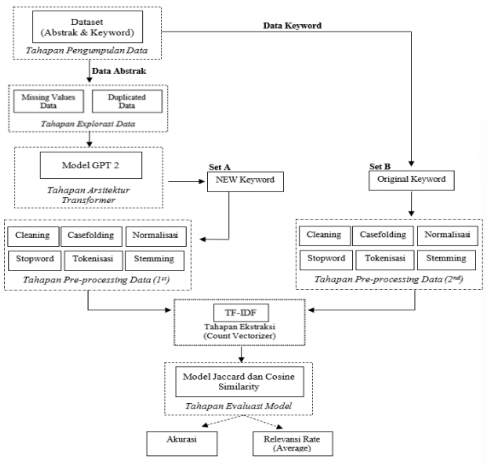


Figure 1. Research Stages.
Source: (Research Results, 2025)

2.1. Data Collection

At this stage, the data used are derived from a secondary dataset consisting of a collection of research article abstracts and their corresponding keywords. The dataset is the publicly available IEEE Xplore Paper Details dataset obtained from Kaggle (https://www.kaggle.com/datasets/naifislam/ieeexplore-paper-details). It contains 174,975 English-language abstracts of articles indexed in IEEE Xplore, in the field of computer science and computing engineering, together with the keywords assigned to each article. As shown by the sample records in Table 1 and Figure 2, the topics include deep learning and image recognition, evolutionary and swarm-based optimization, object detection, and computer architecture. Because generating keywords with GPT-2 for every abstract requires substantial computation time, this research does not process the entire dataset. Instead, a random sample of 5,667 abstract–keyword pairs was drawn from the 174,975 records and used as the research data. This sample size is about 3.2% of the dataset and is still large enough to describe the distribution of similarity scores between generated and original keywords. The data are organized into a single dataset containing two columns: the Abstract column and the Keywords column. The Abstract column serves as the main feature to be processed by the GPT-2 model, while the Keywords column functions as the target reference used to evaluate the model's performance in generating relevant keywords.

2.2. Data Exploration

The stages of data exploration in NLP that involve missing values and duplicated data aim to ensure data quality before it is used in model training or analysis. Missing-value checking is used to identify text data that are empty or contain NaN values, while duplicate checking is used to detect entries that are identical to other entries in the dataset.

2.3. Keyword Generation with GPT-2

At this stage, new keywords are generated from each raw abstract by combining three components. First, Named Entity Recognition (NER) with the spaCy library identifies named entities and technical terms in the abstract, such as method names, dataset names, competitions, and organizations, as keyword candidates. Second, the pre-trained

GPT-2 model receives the abstract as input and generates additional candidate keywords based on the context of the whole text. Third, TF-IDF scores are computed for the words of the abstract to measure the importance of each candidate. The candidates from NER and GPT-2 are then merged, duplicate terms are removed, and the candidates are ranked by their TF-IDF importance to form the final New Keywords of each abstract (Table 2). GPT-2 thus acts as a generative model that uses the context of the whole abstract, while NER and TF-IDF keep the selected keywords grounded in the terms that actually appear in the text.

2.4. Data Pre-processing

Data pre-processing in NLP is a step to prepare raw text into a format that can be understood by the model. Here is an explanation of the main stages of pre-processing, which include cleaning, casefolding, normalization, tokenization, stopword removal, and stemming. The goal is to clean, simplify, and streamline the text so that the NLP model can work more efficiently and accurately. Pre-processing is an important foundation in the NLP pipeline before entering the analysis or model training stage [18], [19].

Data pre-processing is a step to clean and prepare the data before the word weighting and similarity calculation processes. This stage ensures that the data is more structured and ready to be used in subsequent analyses [20]. In this research, pre-processing is applied after the GPT-2 stage, and only to the keywords generated by GPT-2 and to the original author keywords. The abstracts are fed into GPT-2 as raw text, without normalization, stopword removal, or stemming, and are tokenized by the GPT-2 byte-pair encoding (BPE) tokenizer, so that the context required by the transformer is preserved [18]. This stage consists of the following steps:

a. The cleaning stage is carried out to remove unnecessary characters.
b. The casefolding stage converts the entire text to lowercase.
c. The normalization stage replaces non-standard words with standard words.
d. The tokenization stage separates sentences into individual words.
e. The stopword removal stage eliminates words that do not have significant meaning.
f. The stemming stage transforms affixed words into root words.

2.5. TF-IDF Feature Extraction

Feature extraction in Natural Language Processing (NLP) aims to convert text into a numerical representation that can be processed by machine learning algorithms [21]. The method used is TF-IDF (Term Frequency - Inverse Document Frequency), implemented with the TfidfVectorizer class of the scikit-learn library. The feature extraction stage can also identify terms or attributes in text documents. This process is important in data analysis to identify significant words. After the tokenization process separates the text into individual words, these words are converted into statistical values to make them easier to count. TF-IDF measures how important a word is in a document compared to its occurrence in the entire collection of documents [22]. The formula for TF (Term Frequency) is as follows:

$$\boldsymbol{TF(t,d)} = \frac{\boldsymbol{f_{t,d}}}{\boldsymbol{|d|}} \quad (1)$$

where $f_{t,d}$ is the number of occurrences of term t in document d and |d| is the total number of terms in document d.

The formula for IDF (Inverse Document Frequency) is as follows:

$$\boldsymbol{IDF(t)} = \boldsymbol{log}\left(\frac{\boldsymbol{N}}{\boldsymbol{df(t)}}\right) \quad (2)$$

where N is the total number of documents (keyword texts), and df(t) is the number of documents that contain term t. In the implementation, the scikit-learn TfidfVectorizer uses the smoothed form IDF(t) = ln((1 + N)/(1 + df(t))) + 1 and normalizes each TF-IDF vector to unit length (L2 norm).

The TF-IDF weight is then calculated as follows:

$$\boldsymbol{TF-IDF(t,d)} = \boldsymbol{TF(t,d)} \times \boldsymbol{IDF(t)} \quad (3)$$

2.6. Model Evaluation

The keywords generated by the GPT-2 model are then compared with the original

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (173-184)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.54838

author keywords in the Keywords column using Jaccard Similarity and Cosine Similarity. These results are then used to analyze the model for generating keywords for the abstracts of research articles, which is measured in terms of accuracy and relevance based on the average similarity distance [1]. In Equations (4) and (5), A denotes the keywords generated by GPT-2 and B denotes the original author keywords, represented as sets of tokens for Jaccard Similarity and as TF-IDF vectors for Cosine Similarity.

$$\boldsymbol{Jaccard}\ (\boldsymbol{A},\boldsymbol{B}) = \frac{|\boldsymbol{A} \cap \boldsymbol{B}|}{|\boldsymbol{A} \cup \boldsymbol{B}|} \quad (4)$$

$$\boldsymbol{Cosine}\ (\boldsymbol{A},\boldsymbol{B}) = \frac{|\boldsymbol{A}\ .\boldsymbol{B}|}{|\boldsymbol{A}\ |\ |\boldsymbol{B}|} \quad (5)$$

Jaccard and Cosine Similarity produce continuous scores in the range 0–1, so these scores are not classification accuracy by themselves. In this research, each keyword pair is converted into a binary decision using a similarity threshold τ: a pair is classified as matched (True) if its similarity score is greater than or equal to τ, and as not matched (False) otherwise. Accuracy is then defined as the percentage of keyword pairs classified as matched:

$$\boldsymbol{Accuracy} = \frac{\mathbf{1}}{\boldsymbol{N}} \sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{N}} \boldsymbol{I}(\boldsymbol{sim_i} \geq \boldsymbol{\tau}) \times \mathbf{100\%} \quad (6)$$

where N is the number of evaluated keyword pairs (5,667), sim_i is the Jaccard or Cosine Similarity score of pair i, and I(·) is the indicator function, which equals 1 if the condition is true and 0 otherwise. The threshold τ is selected by testing values from 0.1 to 0.9 (Table 4). In addition, the average similarity score of each method is reported separately as a measure of average relevance.

3. RESULTS AND DISCUSSION

The results and analysis of each stage conducted in this research are presented below.

3.1. Data Collection

At this stage, the research uses secondary data, which is data that has been collected and published previously by others for specific purposes, but can be reused in this research. Sources of secondary data can come from scientific journals, books, research reports, government data, or publicly available databases. Secondary data is useful for supporting analysis, comparing results, and enriching insights without the need for direct data collection. In this research, secondary data plays an important role in supporting the analysis and comparison of results. The data used were obtained from Kaggle.com through the following link:

https://www.kaggle.com/datasets/naifislam/ieeexplore-paper-details.

Table 1. Dataset Sample

Title	link	abstract	keywords
Deep Residual Learning for Image Recognition	https://ieeexplore.ieee.org/document/7780459/	Deeper neural networks are more difficult to t...	['Training', 'Degradation', 'Complexity theory..
A fast and elitist multiobjective genetic algo...	https://ieeexplore.ieee.org/document/996017/	Multi-objective evolutionary algorithm	['Genetic algorithms', 'Sorting'
Particle swarm optimization	https://ieeexplore.ieee.org/document/488968/	A concept for the optimization of nonlinea	['Particle swarm optimization', 'Birds', 'Educ...

Source: (Research Results, 2025)

Table 1 shows the dataset downloaded from Kaggle.com in CSV format for analysis purposes. The CSV (Comma-Separated Values) format is used to store data in tabular form, where each row represents a single record, and columns are separated by commas or other delimiters. This dataset contains 174,975 research journal abstracts and will be used as secondary data in the analysis process. The dataset in the CSV file consists of four features: Title (Research Title), Link (Journal Access Page), Abstract (Research Abstract), and Keywords (Research Keywords). However, this research only uses the Abstract and Keywords features because both are relevant for text or word matching.

3.2. Data Exploration

After the dataset is collected, the next step is data exploration to select and understand the characteristics of the data before the keyword extraction process. This process includes determining the features in the content column that will be processed, as well as

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (173-184)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.54838

cleaning data that contains missing values and duplicated data. Missing values are data that are missing or empty, making them impossible to fill in accurately, while duplicated data are data that have similarities in one or more columns. Therefore, data exploration needs to be conducted to ensure that the analysis results are more valid and representative. The exploration was carried out on the 5,667 sampled abstract–keyword pairs, and no missing values or duplicated entries were found in the Abstract and Keywords columns, so all 5,667 pairs were retained for the next stage.

3.3. Keyword Generation with GPT-2

After the exploration phase of checking for missing values and duplicated data, the 5,667 sampled abstract–keyword pairs are confirmed to be free from empty and duplicate data. At the keyword generation stage, the author uses a model with GPT-2 as a generative model to extract new keywords from research abstracts in previously explored secondary data. This model analyzes natural language patterns to generate more relevant and meaningful keywords to support research analysis. The extraction process is carried out using a combination of three methods, namely Named Entity Recognition (NER), GPT-2, and TF-IDF. NER uses spaCy to identify important entities in the text, such as names, locations, and organizations [23]. GPT-2 from OpenAI is used to generate candidate keywords based on the context of the abstract. Meanwhile, TF-IDF is applied to assess the importance level of a word in the abstract based on its frequency compared to other documents and is also used to calculate the importance level of a word in the document. Table 2 displays the results of the keyword generation stage using GPT-2.

Table 2. Results of the Keyword Generation Stage with GPT-2

abstract	NEW keywords
Deeper neural networks are more difficult to t...	['up to', 'ILSVRC & COCO 2015 competitions1', ..
Multi-objective evolutionary algorithms (MOEAs...	['MOEA', 'Sorting Genetic Algorithm II', 'five...
A concept for the optimization of nonlinear fu..	['swarm', 'particle', 'described', 'paradigms'..

Source: (Research Results, 2025)

The combination of these three methods aims to improve the quality of new keyword selection, so that the results obtained are more representative of the abstract's content. The keywords generated through this method will subsequently be further processed in the data pre-processing stage to ensure that the data is in optimal condition before being analyzed in depth.

3.4. Data Pre-processing

This stage is the data preparation process to facilitate the analysis and modeling that will be used. New abstract data (NEW keywords) generated through the GPT-2 technique will be further processed in the pre-processing stage to ensure data quality and consistency before being used in analysis. The data pre-processing stages are as follows:

a. Cleaning. This stage aims to clean the data from errors, inconsistencies, and irrelevant information. This process includes the removal of URLs, HTML, emojis, symbols, numbers, and usernames in the New Keywords column generated by GPT-2.
b. Casefolding. This process converts the entire text to lowercase to ensure consistency, allowing for more accurate analysis.
c. Normalization. This process is carried out to convert the text into a standard form and reduce redundancy. The technique used is lemmatization with WordNetLemmatizer from the NLTK library, which converts words to their base forms according to the linguistic dictionary.
d. Tokenization. This process divides the text into small units called tokens (usually in the form of words). Each word separated by a space is considered one token.
e. Stopword Removal. This process removes common words that frequently appear but do not have significant meaning in the analysis, such as "the," "is," and "and." By eliminating stopwords, the text becomes more concise and relevant, thereby increasing the efficiency of the analysis.
f. Stemming. This process aims to remove word endings to obtain the base form (stem), thereby reducing word variations. This process helps simplify the text and enhance the effectiveness of analysis, especially in feature extraction and semantic understanding.

Table 3 displays the results of the pre-processing stage on the New Keywords generated by the GPT-2 technique in this study.

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (173-184)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.54838

Table 3. Results of the Data Pre-Processing Stage on New Keywords

(Cleaning)	**(Casefolding)**
up to ILSVRC COCO competitions nets VGG Im... MOEA Sorting Genetic Algorithm II five three	up to ilsvrc coco competitions nets vgg im... moea sorting genetic algorithm ii five three
(Normalization)	**(Tokenization)**
up to ilsvrc coco competition net vgg imagenet... moea sorting genetic algorithm ii five three	[up, to, ilsvrc, coco, competition, net, vgg, ... [moea, sorting, genetic, algorithm, ii,
(Stopword Removal)	**(Stemming)**
[ilsvrc, coco, competition, net, vgg, imagenet... [moea, sorting, genetic, algorithm, ii,	ilsvrc coco competition net vgg imagenet resid... moea sorting genetic algorithm ii five three

Source: (Research Results, 2025)

Next, the initial keywords column in the dataset also undergoes the same pre-processing. The data is checked to ensure there are no missing values or duplicate data that could cause bias. This step aims to standardize the format of the initial keywords with the keywords extracted by GPT-2. With uniform pre-processing, accuracy analysis using Jaccard Similarity and Cosine Similarity can be conducted more accurately and objectively. Without this step, differences in text format such as capitalization, word variations, or irrelevant symbols can make the calculation results biased or invalid. Therefore, pre-processing is important so that the evaluation of keyword similarity truly reflects the meaning and relevance in the research abstract. Accordingly, in this research, the pre-processing stage was conducted in two distinct phases.

3.5. TF-IDF Feature Extraction

After pre-processing is applied to the new keywords (New Keywords) and the initial keywords (Old Keywords), the next step is feature extraction using TF-IDF. Feature extraction is important in text modeling and NLP because it converts text into numbers so that it can be processed by machine learning algorithms. For each keyword pair, the TfidfVectorizer class is applied to the two texts (text 1, the old keywords, and text 2, the new keywords) with the fit_transform() method, so that each word is weighted according to Equations (1)–(3). Because the vectorizer is fitted separately on each pair, N = 2: words that appear in both keyword texts receive a lower IDF weight than words that appear in only one of them. The resulting Cosine score therefore mainly reflects the weighted overlap between the two keyword sets, rather than the rarity of each word across the whole dataset. Cosine Similarity is then calculated from the two TF-IDF vectors by measuring the angle between them in the multidimensional feature space; the higher the score, the more similar the two keyword sets are. This technique is useful for assessing the relevance of the generated keywords to the original keywords of the research abstracts.

3.6. Model Evaluation

After feature extraction, text similarity is calculated using Jaccard Similarity and Cosine Similarity. Both methods are used to measure the similarity level between text 1 (old keywords) and text 2 (new keywords) generated through the GPT-2 model. Jaccard Similarity is calculated directly on the sets of preprocessed tokens, without TF-IDF weighting, as the ratio of the number of identical words in both texts to the total number of unique words contained within them. Meanwhile, Cosine Similarity measures similarity based on the angle between two vectors in the TF-IDF feature space, which represent the distribution of words in the document. Figure 2 displays the accuracy calculation results using Jaccard Similarity and Cosine Similarity, as well as the average score of both methods. These results illustrate the level of similarity between the old keywords and the newly generated keywords.

No.	Keyword Lama (Bersih)	Keyword Baru (NEW_KEYWORD)	Jaccard Similarity	Cosine Similarity
1	training, degradation, complexity, theory, image, classification	ilsvrc, competition, imagenet, residual, image, recognition	0.000	0.000
2	genetic, algorithm, sorting, computational, complexity	sorting, genetic, algorithm, problem, pareto, solution	0.158	0.138
3	particle, swarm, optimization, bird, education	particle, described, paradigm, optimization	0.100	0.203
4	computer, architecture, convolutional, code, neural	imagenet, largescale, visual, recognition, challenge	0.095	0.075
5	computer, architecture, microprocessor, object, code	object, detection, second, network	0.250	0.310

Figure 2. Display of Similarity Results Using Jaccard Similarity and Cosine Similarity Methods.
Source: (Research Results, 2025)

Figure 2 illustrates a table displaying sample results of Jaccard and Cosine Similarity calculations between pairs of old and new keywords in scientific literature. The columns keyword_lama_bersih and NEW_KEYWORD contain preprocessed keyword tokens, while the last two columns present the corresponding similarity values. It is evident that Cosine Similarity generally produces higher scores, indicating that the weighted vector

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (173-184)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.54838

representation is more tolerant of partial overlap between keyword sets. The total number of analyzed keyword pairs reaches 5,667, which strengthens the validity of the results.

Figure 3 displays the resulting similarity histogram.

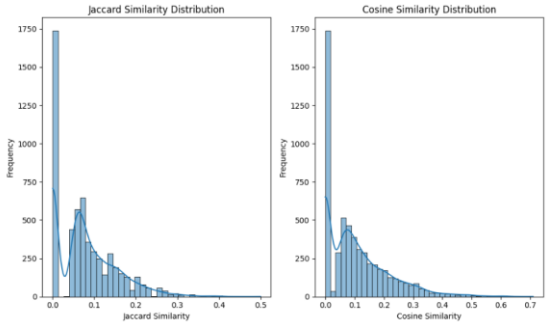


Figure 3. Histogram of Jaccard and Cosine Similarity Results
Source: (Research Results, 2025)

Figure 3 presents a histogram showing the frequency distribution of Jaccard and Cosine similarity scores obtained from the comparison between old and new keywords in scientific documents. Most keyword pairs exhibit very low similarity levels, particularly in the 0.0 to 0.1 range, with the highest frequency exceeding 1,700 instances. The Jaccard similarity distribution appears heavily concentrated around zero because it accounts only for direct token overlap, resulting in low scores when no explicit matches are found. On the other hand, Cosine similarity values show a broader spread, reaching up to approximately 0.7, indicating that this approach is more flexible in recognizing partial matches through vector-based weighting methods such as TF-IDF, even when the words are not lexically identical. This difference suggests that Cosine Similarity is more tolerant than Jaccard Similarity in recognizing partially matching keywords, although both measures remain based on shared words rather than on meaning. These findings emphasize the importance of developing automated keyword adjustment systems using NLP techniques to improve the accuracy of indexing processes and enhance information accessibility in academic literature.

Figure 4 displays the resulting scatter plot of the correlation between the similarity scores.

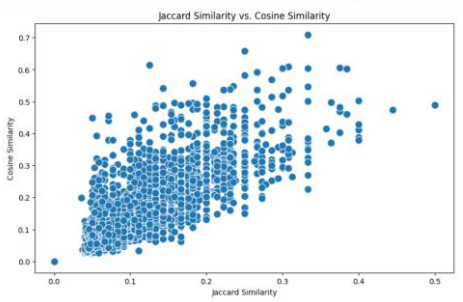


Figure 4. Scatter Plot of the Correlation between Jaccard and Cosine Similarity
Source: (Research Results, 2025)

Figure 4 presents the correlation between Jaccard Similarity and Cosine Similarity in evaluating the similarity of scientific keyword pairs. A moderate positive trend is observed: as Jaccard values increase, Cosine values generally rise, although the relationship is not strictly linear. Most data points are concentrated in the lower similarity range, indicating that the majority of keyword pairs share limited lexical and semantic overlap. However, Cosine Similarity tends to produce higher scores than Jaccard, because the TF-IDF weighting and length normalization give credit to partially overlapping keyword sets, whereas Jaccard counts only exact shared tokens. These findings highlight the importance of selecting an appropriate similarity metric in the development of automated keyword alignment systems using NLP, particularly to enhance indexing and information retrieval in scientific literature.

Figure 5 displays a colored scatter plot representing the correlation pattern of the computed average similarity.

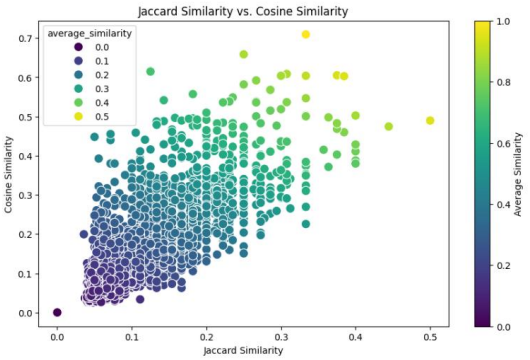


Figure 5. Colored Scatter Plot of Average Similarity
Source: (Research Results, 2025)

Figure 5 presents a colored scatter plot illustrating the relationship between Jaccard

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (173-184)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.54838

and Cosine Similarity values for keyword pairs in scientific literature, with color intensity representing the average similarity. The color of each data point indicates the average level of similarity—yellow denotes high similarity, while purple indicates low similarity. A positive correlation is observed between the two metrics, with Cosine Similarity generally producing higher values than Jaccard. This visualization shows that the average similarity is driven mainly by the Cosine scores.

The histogram of the similarity score distribution for Jaccard Similarity and Cosine Similarity shows different patterns in measuring data similarity. In Jaccard Similarity, most scores fall within the range of 0 to 0.5, with the majority approaching 0. This indicates that the compared data have low similarity because Jaccard only counts the parts that are truly the same compared to the entire combined data. Meanwhile, Cosine Similarity has a wider range of scores, up to about 0.7, with the peak value at 0. This means that data vectors tend to be more similar in multidimensional space, even though the size (magnitude) of the vectors may differ. This difference indicates that Jaccard Similarity is stricter in measuring similarity, while Cosine Similarity is more flexible. The choice of metric depends on the type of relationship or pattern that one wants to analyze in the data.

After determining the similarity results from both methods, namely Jaccard and Cosine, as well as the average similarity, this research determines the threshold value τ used in Equation (6). The threshold search was first run with GridSearchCV and a simple ThresholdClassifier model that compares scores against τ to determine the class. However, because the dataset has no ground-truth labels indicating whether a keyword pair is truly relevant, GridSearchCV could not compute a valid score (Best score: nan, Figure 6), and its selection of 0.1 only corresponds to the first value in the search grid. The threshold was therefore determined through a sensitivity analysis, as presented in Table 4.

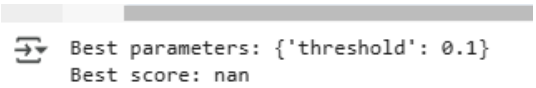


Figure 6. GridSearchCV Output for the Threshold Parameter.
Source: (Research Results, 2025)

In the process of determining the best parameter for the similarity threshold, various threshold values were tested to identify a suitable parameter for matching keyword pairs as similar or not. The testing was conducted using threshold values ranging from 0.1 to 0.9, and for each value the accuracy was calculated with Equation (6) using the average of the Jaccard and Cosine Similarity scores. The following table presents the results of the parameter testing for threshold determination.

Table 4. Threshold Sensitivity Testing

Threshold	Accuracy (Average Similarity)
0.1	54.80%
0.2	12.40%
0.3	0.70%
0.4	0.00%
0.5	0.00%
0.6	0.00%
0.7	0.00%
0.8	0.00%
0.9	0.00%

Source: (Research Results, 2025)

Table 4 shows that accuracy decreases sharply as the threshold increases and reaches 0.00% from $\tau = 0.4$ onwards, because most keyword pairs have low similarity scores (Figure 3). Although individual Cosine scores can reach approximately 0.7, pairs with high Cosine scores rarely also have high Jaccard scores, so no pair reaches an average similarity of 0.4 or higher. The threshold $\tau = 0.1$ is the only tested value that still classifies a meaningful share of keyword pairs as matched, so it is applied to both methods. Since 0.1 is the lowest value in the tested range, thresholds below 0.1 were not evaluated in this study and are left for future work. At this threshold, the accuracy based on the average similarity is 54.80%, while the accuracy of each individual method is shown in Figure 7.

Figure 7 displays the accuracy results at the 0.1 threshold for both methods:

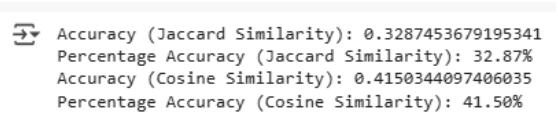


Figure 7. Display of Jaccard and Cosine Similarity Method Accuracy Results (Threshold = 0.1)
Source: (Research Results, 2025)

Additionally, Figures 8 and 9 show the previously found accuracy results, which can be

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (173-184)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.54838

classified as True and False prediction classifications based on their frequency:

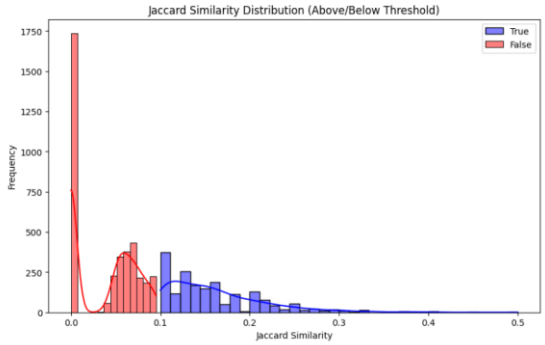


Figure 8. Classification Frequency Display in Jaccard Similarity Model Prediction
Source: (Research Results, 2025)

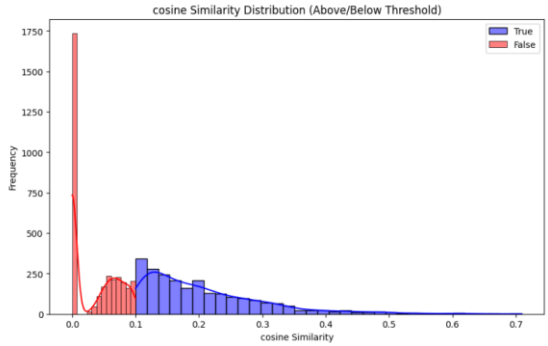


Figure 9. Classification Frequency Display in Cosine Similarity Model Prediction
Source: (Research Results, 2025)

Figures 8 and 9 illustrate the frequency distribution patterns of keyword pairs based on Jaccard Similarity and Cosine Similarity scores, with classification determined by a threshold value of 0.1. It is evident that most frequencies are concentrated below the 0.1 similarity score (highlighted in red), indicating low literal similarity among the pairs. Meanwhile, pairs with similarity scores equal to or above 0.1 (shown in blue) tend to be more evenly distributed, although they occur less frequently. This pattern confirms that the majority of keyword pairs do not exhibit explicit token matches, and that the 0.1 threshold can be considered an appropriate parameter for similarity classification using token-based methods.

These accuracy values should be interpreted in light of how author keywords are formed. Lu et al. [5] found that, on average, only about 56.7% of author keywords appear in the title and abstract of a paper, while the others come from the authors' prior knowledge and the background of the research field. Because the GPT-2, NER, and TF-IDF pipeline in this research uses only the abstract as input, a substantial share of the original keywords cannot be recovered from the text. This places a natural upper limit on the achievable similarity and explains why most keyword pairs have scores below 0.1 (Figures 3, 8, and 9). Within this limit, the 41.50% of pairs that reach the Cosine threshold suggests that the generated keywords capture a considerable part of the content-based author keywords. Previous comparisons of unsupervised keyword extraction tools also show that performance varies with text length and data type, with the embedding-based KeyBERT achieving the highest F-score and statistical tools such as YAKE and RAKE being the fastest [8]. A direct comparison with these tools on the same dataset is therefore needed to position the proposed approach more precisely.

CONCLUSION

This research shows that a keyword generation pipeline based on GPT-2, combined with Named Entity Recognition (NER) and Term Frequency–Inverse Document Frequency (TF-IDF) techniques, can generate keywords that partly match the author keywords of the research abstracts in the dataset. The evaluation results indicate that Cosine Similarity yields a higher proportion of matched pairs, with an accuracy of 41.50%, compared to Jaccard Similarity, which achieves 32.87%, where accuracy is the percentage of keyword pairs whose similarity score reaches the threshold (Equation 6). These values are bounded by the fact that a substantial share of author keywords does not appear in the abstract itself [5]. The accuracy was determined by applying a threshold value of 0.1, selected through a sensitivity analysis of thresholds from 0.1 to 0.9. The main contribution of this research lies in the empirical evaluation of a Natural Language Processing (NLP) keyword generation pipeline that integrates Generative Pre-trained Transformer 2 (GPT-2) with NER and TF-IDF methods, and in showing how the choice between lexical and vector-based similarity metrics affects the assessment of generated keywords. The results indicate that this combination can produce keywords that partly match the author keywords; however, because no baseline extractor was evaluated,

this research does not yet show that it outperforms other keyword extraction methods. This research has several limitations. The dataset has no ground-truth labels of keyword relevance, so the reported accuracy is not a classification accuracy against verified labels, but the proportion of keyword pairs whose similarity score reaches the threshold. In addition, only thresholds from 0.1 to 0.9 were tested; the TF-IDF vectorizer was fitted per keyword pair rather than on the whole corpus, so the IDF weights do not reflect word rarity across the dataset; and the GPT-2 keywords were not compared with other keyword extraction methods. For future research, it is recommended to fit the TF-IDF weights on the entire keyword corpus, to build an evaluation set with human-annotated keyword relevance, to compare the model with established keyword extractors such as YAKE and KeyBERT [8], and to explore more advanced transformer models, such as GPT-3, BERT, or T5, and to integrate them with contextual word embedding techniques such as Word2Vec, FastText, or Sentence-BERT. Such exploration may help capture semantic relationships between keywords that do not share the same words, which the current lexical measures cannot detect.

REFERENCES

[1] Singh, R. and Singh, S. (2021) 'Text Similarity Measures in News Articles by Vector Space Model Using NLP', Journal of The Institution of Engineers (India): Series B, 102(2), pp. 329–338. Available at: https://doi.org/10.1007/s40031-020-00501-5.

[2] Khurana, D. et al. (2023) 'Natural language processing: state of the art, current trends and challenges', Multimedia Tools and Applications, 82(3), pp. 3713–3744. Available at: https://doi.org/10.1007/s11042-022-13428-4.

[3] Montejo-Ráez, A. and Jiménez-Zafra, S.M. (2022) 'Current Approaches and Applications in Natural Language Processing', Applied Sciences, 12(10), p. 4859. Available at: https://doi.org/10.3390/app12104859.

[4] Galletti, M. et al. (2025) 'Are Your Keywords Like My Queries? A Corpus-Wide Evaluation of Keyword Extractors with Real Searches', Proceedings - International Conference on Computational Linguistics, COLING, Part F2064, pp. 1943–1951.

[5] Lu, W., Liu, Z., Huang, Y., Bu, Y., Li, X. and Cheng, Q. (2020) 'How do authors select keywords? A preliminary study of author keyword selection behavior', Journal of Informetrics, 14(4), p. 101066. Available at: https://doi.org/10.1016/j.joi.2020.101066.

[6] Singh, V.K. et al. (2021) The journal coverage of Web of Science, Scopus and Dimensions: A comparative analysis, Scientometrics. Springer International Publishing. Available at: https://doi.org/10.1007/s11192-021-03948-5.

[7] Qian, Y., Jia, C. and Liu, Y. (2021) 'Bert-Based Text Keyword Extraction', Journal of Physics: Conference Series, 1992(4). Available at: https://doi.org/10.1088/1742-6596/1992/4/042077.

[8] Nadim, M., Akopian, D. and Matamoros, A. (2023) 'A Comparative Assessment of Unsupervised Keyword Extraction Tools', IEEE Access, 11, pp. 144778–144798. Available at: https://doi.org/10.1109/ACCESS.2023.3344032.

[9] Sufi, F. (2024) 'Generative Pre-Trained Transformer (GPT) in Research: A Systematic Review on Data Augmentation', Information, 15(2), p. 99. Available at: https://doi.org/10.3390/info15020099.

[10] Yenduri, G. et al. (2024) 'GPT (Generative Pre-Trained Transformer)—A Comprehensive Review on Enabling Technologies, Potential Applications, Emerging Challenges, and Future Directions', IEEE Access, 12(March), pp. 54608–54649. Available at: https://doi.org/10.1109/ACCESS.2024.3389497.

[11] Fröhling, L. and Zubiaga, A. (2021) 'Feature-based detection of automated language models: Tackling GPT-2, GPT-3 and Grover', PeerJ Computer Science,

7, pp. 1–23. Available at: https://doi.org/10.7717/PEERJ-CS.443.

[12] Kumar, A., Sharma, R. and Bedi, P. (2024) 'Towards Optimal NLP Solutions: Analyzing GPT and LLaMA-2 Models Across Model Scale, Dataset Size, and Task Diversity', Engineering, Technology and Applied Science Research, 14(3), pp. 14219–14224. Available at: https://doi.org/10.48084/etasr.7200.

[13] Milani Fitria, K. (2023) 'Information Retrieval Performance in Text Generation using Knowledge from Generative Pre-trained Transformer (GPT-3)', Jambura Journal of Mathematics, 5(2), pp. 327–338. Available at: https://doi.org/10.34312/jjom.v5i2.20574.

[14] Akuma, S., Lubem, T. and Adom, I.T. (2022) 'Comparing Bag of Words and TF-IDF with different models for hate speech detection from live tweets', International Journal of Information Technology (Singapore), 14(7), pp. 3629–3635. Available at: https://doi.org/10.1007/s41870-022-01096-4.

[15] Egger, R. (2022) 'Text Representations and Word Embeddings: Vectorizing Textual Data', Tourism on the Verge, Part F1051(January 2022), pp. 335–361. Available at: https://doi.org/10.1007/978-3-030-88389-8_16.

[16] Mishra, P. et al. (2021) 'Automatic Title Generation for Text with Pre-trained Transformer Language Model', in 2021 IEEE 15th International Conference on Semantic Computing (ICSC). IEEE, pp. 17–24. Available at: https://doi.org/10.1109/ICSC50631.2021.00009.

[17] Perez-Castro, A., Martínez-Torres, M.R. and Toral, S.L. (2023) 'Efficiency of automatic text generators for online review content generation', Technological Forecasting and Social Change, 189(January 2022), p. 122380. Available at: https://doi.org/10.1016/j.techfore.2023.122380.

[18] Siino, M., Tinnirello, I. and La Cascia, M. (2024) 'Is text preprocessing still worth the time? A comparative survey on the influence of popular preprocessing methods on Transformers and traditional classifiers', Information Systems, 121, p. 102342. Available at: https://doi.org/10.1016/j.is.2023.102342.

[19] Chai, C.P. (2023) 'Comparison of text preprocessing methods', Natural Language Engineering, 29(3), pp. 509–553. Available at: https://doi.org/10.1017/S1351324922000213.

[20] Prasetyo, V.R., Erlangga, G. and Prima, D.A. (2023) 'Analisis Sentimen untuk Identifikasi Bantuan Korban Bencana Alam berdasarkan Data di Twitter Menggunakan Metode K-Means dan Naive Bayes', Jurnal Teknologi Informasi dan Ilmu Komputer, 10(5), pp. 1055–1062. Available at: https://doi.org/10.25126/jtiik.2023105707.

[21] Hassan, S.U., Ahamed, J. and Ahmad, K. (2022) 'Analytics of machine learning-based algorithms for text classification', Sustainable Operations and Computers, 3(February), pp. 238–248. Available at: https://doi.org/10.1016/j.susoc.2022.03.001.

[22] Tarigan, D.A., Situmorang, Z. and Rosnelly, R. (2025) 'Analisis Sentimen Aplikasi Playstore Sirekap 2024 Pasca Pilpres dengan Perbandingan Metode Support Vector Machine (SVM), Naïve Bayes Classifier dan Random Forest', Jurnal Teknologi Informasi dan Ilmu Komputer, 12(3), pp. 661–670.

[23] Sagheb, E. et al. (2021) 'Use of Natural Language Processing Algorithms to Identify Common Data Elements in Operative Notes for Knee Arthroplasty', Journal of Arthroplasty, 36(3), pp. 922–926. Available at: https://doi.org/10.1016/j.arth.2020.09.029.