International Journal of Electrical and Computer Engineering (IJECE)
Vol. 15, No. 6, December 2025, pp. 5543~5554
ISSN: 2088-8708, DOI: 10.11591/ijece.v15i6.pp5543-5554

Optimizing parameter selection in bidirectional encoder portrayal for transformers algorithm using particle swarm optimization for artificial intelligence generate essay detection

Tegar Arifin Prasetyo[1], Rudy Chandra[1], Wesly Mailander Siagian[2], Horas Marolop Amsal Siregar[1], Samuel Jefri Saputra Siahaan[1]
[1]Department of Information Technology, Faculty of Vocational Studies, Institut Teknologi Del, Toba, Indonesia
[2]Department of Engineering Management, Faculty of Industrial Technology Faculty, Institut Teknologi Del, Toba, Indonesia

Article Info

Article history:

Received Nov 3, 2024
Revised Jul 16, 2025
Accepted Sep 14, 2025

Keywords:

Artificial intelligence generates essay detection
Bidirectional encoder portrayal for transformers algorithm
Particle swarm optimization algorithm
Natural language processing
Optimization parameter

ABSTRACT

This research proposes a novel method for detecting artificial intelligence (AI)-generated essays by integrating the bidirectional encoder representations from transformers (BERT) model with particle swarm optimization (PSO). Unlike traditional approaches that rely on manual hyperparameter tuning, this study introduces a systematic optimization technique using PSO to improve BERT's performance in identifying AI-generated content. The key problem addressed is the lack of effective, real-time detection systems that preserve academic integrity amidst rapid AI advancements. This optimization enhances the model's detection accuracy and operational efficiency. The research dataset consisted of 46,246 essays, which, after data cleaning, were refined to 44,868. The model was then tested on 9,250 essays. Initial evaluations showed BERT's accuracy ranging from 83% to 94%. After being optimized with PSO, the model achieved an accuracy of 98%, an F1-score of 98.31%, precision of 97.75%, and recall of 98.87%. The model was deployed using a FastAPI-based web interface, enabling real-time detection and providing users with an efficient way to quickly verify text authenticity. This research contributes a scalable, automated solution for AI-generated text detection and offers promising implications for its application in various academic and digital content verification contexts.

Corresponding Author:

Tegar Arifin Prasetyo
Department of Information Technology, Faculty of Vocational Studies, Institut Teknologi Del
Sisingamangaraja Street, Sitoluama, Laguboti 22381, Toba, Indonesia
Email: tegar.prasetyo@del.ac.id

1. INTRODUCTION

In recent years, artificial intelligence (AI) has achieved remarkable progress across various sectors. One of the most prominent areas of advancement is in natural language processing (NLP). NLP enables computers to understand, process, and generate human language in a more fluid and efficient way. It is a key technology behind various applications, including virtual assistants and automated text analysis platforms [1]. As progress in this field accelerates, new challenges also surface, one of the main ones being the ability to identify text created by machines [2], [3]. Recent advancements in AI models have enabled the generation of essays and articles that closely mimic human writing, raising significant concerns regarding the authenticity of text, particularly within educational and publishing domains. The need for robust automated detection of machine-generated content has become critical to ensuring the integrity and originality of written materials.

Consequently, the development of effective methodologies to distinguish between human-authored and AI-generated text has emerged as a key research priority in this area [4], [5].

With the rapid advancements in AI, bidirectional encoder representations from transformers (BERT), developed by Google, has emerged as the industry standard due to its exceptional ability to understand and interpret the complex nuances of language. Its bidirectional approach to language comprehension has redefined natural language processing, establishing a new benchmark for performance in the field. BERT has been applied to a wide range of NLP tasks, including text classification [6], sentiment analysis [7], and plagiarism detection [8]. Based on the research conducted by [9], the BERT model demonstrated superior accuracy in sentiment prediction compared to traditional approaches like naïve Bayes, support vector machines (SVM), and long short-term memory network (LSTM). Its pre-trained architecture and ability to fine-tune give BERT an advantage over other techniques, achieving higher performance in metrics such as F1 score, precision, and recall. The research [10] presents a framework that leverages both generative pre-trained transformers (GPT) and BERT for effective fake news detection, achieving 95.30% accuracy in tests conducted on two real-world datasets, which emphasizes its strong potential to address the issue of fake news in the digital era.

Despite its potential, applying BERT for detecting AI-generated essays remains challenging, primarily due to the complexity involved in fine-tuning the model's parameters. This is evidenced by the research, which demonstrates that BERT parameters can be optimized using the whale optimization algorithm for semantic text generation; however, the results still hover around 0.76 on metrics for evaluation of translation, like GPT-2 [11]. The selection of optimal parameters in BERT is critical to maximizing model performance [8]. However, this process is often performed manually or through trial-and-error techniques, which are not only inefficient but also highly time-consuming. BERT relies on a vast array of hyperparameters that critically influence its performance, such as the learning rate, number of layers, embedding size, and batch size [12], [13]. Attaining optimal performance demands a more structured and methodical approach to selecting the most suitable parameters.

This research utilizes a metaheuristic optimization method, namely particle swarm optimization (PSO), to tackle the difficulties associated with optimizing BERT's parameter selection. PSO inspired by the behavior of bird flocks and fish schools, effectively navigates complex search spaces and is useful for tuning hyperparameters in machine learning and deep learning models [14]. The study by [15] suggests that integrating PSO with recursive feature elimination (RFE) for optimizing SVM parameters enhances heart disease detection accuracy from 86.41% to 89.13%, emphasizing PSO's greater effectiveness over traditional methods. In addition, the research by [16] introduces an innovative tiered feature selection technique that leverages PSO alongside genetic algorithm (GA) to optimize the performance of coronary heart disease diagnosis systems. The application of PSO yielded remarkable outcomes, with an accuracy of 96.04%, an AUC of 99.97%, and a sensitivity of 98.36%, highlighting its considerable potential to enhance diagnostic accuracy and reliability in coronary heart disease detection. The use of PSO enables a more efficient and systematic approach to parameter optimization, substantially reducing the reliance on manual experimentation, which is often time-consuming and resource intensive.

The combination of BERT and PSO is expected to significantly enhance the effectiveness of AI-generated essay detection systems. While PSO automates the optimization of parameter configurations, BERT provides a strong foundational model for comprehending the structure and semantics of the text. By applying this method, it is expected that the detection of AI-generated essays will become more accurate and reliable. Moreover, the findings from this study are expected to drive the development of more advanced and context-specific automated detection tools, particularly within the educational sector, where AI is increasingly used to produce academic content like essays. Furthermore, the growing adoption of AI for text generation has raised concerns about academic integrity [17]. In educational institutions, essays generated by AI can obscure the distinction between authentic student work and AI-created content [18]–[20]. Therefore, the development of systems capable of accurately detecting AI-generated essays has become increasingly important to preserve the credibility of academic evaluations. In our research, we suggest that leveraging optimization techniques such as PSO to fine-tune BERT parameters can greatly enhance the speed and accuracy of detecting AI-generated essays, thereby contributing significantly to the preservation of academic integrity. Through this integration, the research aims to provide a more effective solution for improving both the accuracy and efficiency of detecting AI-generated texts. Moreover, our research has the potential to provide fresh insights into systematic strategies for optimizing parameters in deep learning models, which could be extended to a range of future NLP applications. This contribution will expand the current literature on AI advancements in education, particularly in preserving the authenticity of written work in an increasingly digitalized world.

In summary, this research addresses the gap in effective, real-time systems for detecting AI-generated academic content. The novelty lies in combining BERT with PSO for systematic hyperparameter tuning, which

significantly enhances detection accuracy. The implications of this work extend beyond education, suggesting potential applications in journalism, publishing, and content moderation where authenticity is critical.

2. METHOD

Recent advancements in transformer-based models have shown significant improvements in various NLP tasks [21]. BERT has become a foundational model for many applications due to its deep bidirectional encoding. However, alternative models such as RoBERTa, T5, and GPT-series models have emerged, offering improved performance in certain tasks. RoBERTa enhances BERT by removing the Next Sentence Prediction objective and training with more data and larger batches, making it more robust in understanding context [22]. T5, on the other hand, reformulates all NLP tasks into a text-to-text format, showing strong performance on generative tasks [23]. GPT-based models, especially the latest GPT-3 and GPT-4, offer remarkable text generation capabilities and are known for producing human-like responses [24]. Despite their generative strengths, their usage for classification tasks requires fine-tuning and prompt engineering. In comparison, BERT remains a strong choice for binary classification due to its pretrained architecture and attention to contextual semantics. However, fine-tuning BERT effectively remains a challenge due to the vast hyperparameter space. To address this, we implement PSO, a metaheuristic algorithm inspired by the social behavior of birds. PSO has been shown to outperform traditional methods like Grid Search and Random Search in terms of speed and accuracy [25]. It is computationally efficient and suitable for optimizing deep learning model parameters. Alternative optimization methods such as genetic algorithms or Bayesian optimization can be more complex or require more iterations to converge. Therefore, PSO is selected in this study due to its simplicity, effectiveness, and proven success in previous machine learning applications.

Figure 1 illustrates the process of detecting AI-generated essays using BERT optimized with PSO, starting from data input, cleaning, and tokenization. The processed data flows through the BERT model, passing through layers like embedding, self-attention, and feed-forward network, followed by fine-tuning with various hyperparameters. PSO optimization identifies the best hyperparameter combination, and the final prediction—whether the essay is AI-generated or human-written—is evaluated using metrics such as accuracy, precision, recall, and F1 score.

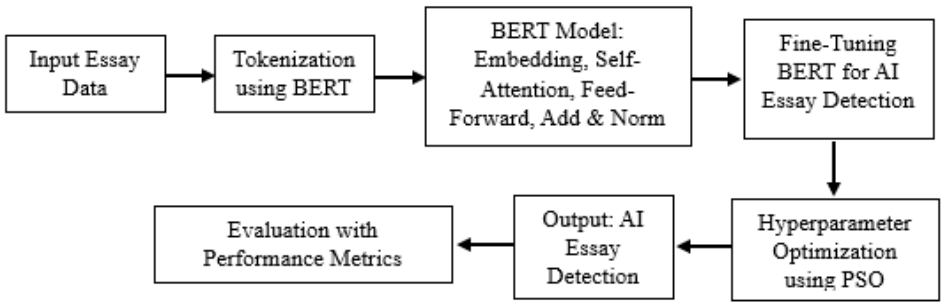


Figure 1. Research workflow diagram

2.1. Data collection and preprocessing

The dataset for implementing the model, sourced from Kaggle and containing 46,246 rows of text for identification, will be split with 80% designated for training and 20% for testing. Then, the data cleaning process began with 46,246 rows, from which 1,378 duplicate entries were removed, resulting in a final dataset of 44,868 rows, with no null values needing removal. The AI-generated samples span multiple domains, including education, technology, general knowledge, and opinion-based writing, offering diverse linguistic structures for robust model training. Examples include:

a. "Artificial Intelligence is transforming education by enabling personalized learning experiences."
b. "The capital of France is Paris."

Preprocessing involved tokenization using the BERT tokenizer, followed by padding and truncation to ensure fixed-length inputs. The processed tokens were then converted into PyTorch tensors for batch processing.

2.2. BERT model architecture

The BERT model used in this research is based on the BertForSequenceClassification architecture. The model includes an embedding layer that transforms input tokens into vectors by incorporating token embeddings, positional encodings, and segment embeddings. These are followed by 12 transformer encoder

layers, each featuring self-attention mechanisms and feed-forward networks. Each encoder layer captures bidirectional contextual relationships among the tokens. The model ends with a pooling layer based on the [CLS] token and a fully connected classification layer that outputs probabilities for two classes: human-written or AI-generated. In total, the architecture contains around 109.5 million parameters, making it highly expressive and capable of handling complex text classification tasks.

2.3. PSO optimization for hyperparameter tuning

Hyperparameter tuning is critical for achieving optimal performance in BERT. In this study, we utilize PSO to automate the tuning process for three key hyperparameters: learning rate, batch size, and number of training epochs. PSO is a population-based optimization algorithm inspired by the collective movement behavior of birds or fish. Each particle in the swarm represents a candidate configuration, and the particles iteratively update their positions based on their own best performance and the global best configuration [26]. The evaluation metric is validation loss, computed during training. We configure PSO to use 10 particles and a maximum of 5 iterations, with early stopping enabled to prevent overfitting. This approach significantly improved the model's accuracy, precision, recall, and F1-score. To enhance the reproducibility of our approach, we provide a clear description of the integration process between BERT and PSO. The input essays first undergo preprocessing and tokenization using the BERT tokenizer, after which they are fed into the BertForSequenceClassification model. The PSO algorithm initializes a population of particles, each representing a possible combination of hyperparameters such as learning rate, batch size, and number of epochs. During training, each particle's configuration is evaluated based on validation loss, and the particles update their positions by considering both their own best performance and the global best solution found so far. This process continues iteratively until an optimal configuration is identified. The best hyperparameter set is then used to fine-tune the BERT model for final evaluation.

2.4. Essay prediction and evaluation

The trained and optimized BERT model is then used to classify essay texts as human-written or AI-generated. Predictions are generated by feeding the processed essay text into the fine-tuned model. The model outputs class probabilities, and the higher-probability class is selected as the prediction result. To facilitate practical usage, the model is deployed using a FastAPI-based web interface. Users can input an essay, and the interface returns an immediate prediction. The effectiveness of this prediction process is validated using metrics such as accuracy, precision, recall, and F1-score, ensuring robust real-time classification capability [27], [28].

3. RESULTS AND DISCUSSION

3.1. Result of tokenization

The tokenization phase converts the input text into a numerical representation optimized for BERT model processing [29]. At this stage, each word is segmented into tokens, either as entire words or sub-words, in alignment with the BERT tokenizer's structure. This segmentation allows uncommon or complex terms to be broken down into smaller units, thereby strengthening the model's ability to interpret and generalize across diverse textual inputs. Subsequently, each text is standardized to a fixed length using padding and truncation based on the specified maximum length. Padding adds blank tokens to reach the desired length, while truncation shortens texts that exceed it. This step is crucial for batch processing, allowing the model to receive inputs of uniform size, thereby streamlining data processing [30]. The tokenized output is subsequently transformed into PyTorch tensors, allowing the model to process the input directly during both training and inference. By structuring the text in tensor format, it is prepared in an optimized numerical form for model computation, which enhances processing speed and organization. This step ensures that the text is consistently and efficiently ready for BERT, enabling the model to concentrate on interpreting the meaning and structure of the text for designated NLP tasks. The results from the BERT tokenizer are presented in Table 1.

Table 1 displays the results of tokenization, providing the numerical representation of the original text prepared for processing by the BERT model. During this process, each text sequence starts with the special token [CLS], indicating the beginning of a sentence, and ends with (SEP), marking the sentence's end or acting as a separator between different text segments. This setup helps the model recognize the boundaries of sentences or paragraphs. Each word or part of a word is converted into a specific token, where complex words or those not found in the vocabulary are broken down into smaller sub-tokens. This process generates a sequence of tokens, each given a unique numeric ID within BERT's vocabulary, such as ID 101 for the special [CLS] token. These IDs allow the model to recognize and handle tokens in a calculable numeric format, ensuring that each word or sub-word maintains a consistent representation during computational

processes. Moreover, to achieve consistent text length, padding and truncation are used. Extra tokens with a value of 0 are appended to shorter texts to meet the desired input length, while texts exceeding the maximum length are trimmed accordingly. The tokenization output also provides an attention mask, guiding the model to attend to relevant tokens (indicated by a value of 1) and to ignore padding tokens (indicated by a value of 0). In this way, the tokenization process not only transforms the text into a numerical format compatible with BERT but also enhances processing efficiency by concentrating on the portions of text that hold relevant information.

Table 1. Tokenized text representations

Original text	Tokens	Token IDs	Attention
"Cars. Cars have been around since they became available to the public..."	[CLS], cars, ., cars, have, been, around, since, they, became, available, to, the, public, ..., [SEP]	[101, 1234, 119, 1234, 2031, 2042, 2235, 2144, 2027, 2150, 2800, 2000, 1996, 2270, 102]	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0]
"Transportation is a large necessity in most countries..."	[CLS], transportation, is, a, large, necessity, in, most, countries, ..., [SEP]	[101, 2036, 2003, 1037, 2312, 4518, 1999, 2087, 3032, 102]	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
"America's love affair with its vehicles seems eternal..."	[CLS], America, 's, love, affair, with, its, vehicles, seems, eternal, ..., [SEP]	[101, 2637, 1521, 1055, 2293, 3941, 2007, 2049, 5872, 4023, 102]	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
"How often do you ride in a car? Do you drive a car often?"	[CLS], how, often, do, you, ride, in, a, car, ?, do, you, drive, a, car, often, ?, [SEP]	[101, 2129, 2411, 2079, 2017, 3854, 1999, 1037, 2482, 1029, 2079, 2017, 3298, 1037, 2482, 2411, 1029, 102]	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0]
…	…	…	…
"Cars are a wonderful thing. They are perhaps one of the greatest inventions in human history."	[CLS], cars, are, a, wonderful, thing, ., they, are, perhaps, one, of, the, greatest, inventions, in, human, history, ., [SEP]	[101, 1234, 2024, 1037, 6919, 2518, 119, 2027, 2024, 3384, 2028, 1997, 1996, 4602, 10508, 1999, 2529, 2381, 119, 102]	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
"Public transportation offers many benefits to urban residents by reducing traffic and pollution levels."	[CLS], public, transportation, offers, many, benefits, to, urban, residents, by, reducing, traffic, and, pollution, levels, ., [SEP]	[101, 2270, 2036, 4161, 2116, 6661, 2000, 3926, 5158, 2011, 11848, 4918, 1998, 9472, 4740, 119, 102]	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0]
"With increased access to public transportation, more people can rely less on personal vehicles."	[CLS], with, increased, access, to, public, transportation, ,, more, people, can, rely, less, on, personal, vehicles, ., [SEP]	[101, 2007, 4122, 3229, 2000, 2270, 2036, 117, 2062, 2111, 2064, 4801, 2629, 2006, 3160, 5872, 119, 102]	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0]

3.2. Result of BERT architecture

Table 2 provides an overview of the BERT model architecture tailored for sequence classification, containing roughly 109.5 million parameters in total. The embedding layer, consisting of word, position, and token-type embeddings, includes 23.9 million parameters and converts text input into vectorized representations. At the model's core, 12 encoder layers house self-attention mechanisms, each with 1.77 million parameters per layer, along with further transformations, amounting to about 85.4 million parameters in total. Following self-attention, each encoder layer includes a dense layer, normalization, and dropout to enhance stability. An intermediate dense layer temporarily expands the dimension to 3072 using GELU activation (requiring 2.36 million parameters) before returning to 768 dimensions. The pooler layer consolidates sentence representation via the [CLS] token, and a final classifier layer with 1,538 parameters generates the class predictions.

In BERT's architecture, every token in a text sequence undergoes multiple transformation stages, enabling the model to understand both the context and the complete meaning of the sentence [31]. Table 3 showcases the BERT process, with a focus on the embedding layer outputs, self-attention operations, and feed-forward network (FFN) stages for each token. Initially, each token passes through an embedding stage, where each word or subword is converted into a 768-dimensional vector. This vector merges details about the token itself, its position in the sentence, and its segment origin. This embedded representation serves as the foundational layer for each token, establishing a preliminary identity for each word within the sentence's context. Following the embedding phase, each token moves into the self-attention stage, where BERT determines the degree of connection between a specific token and every other token within the sentence. This mechanism allows each token to "focus on" other words that are contextually relevant or closely connected to it [32]. As an example, in the phrase "Cars have been around…," the token "cars" may focus more on words like "have" and "been," deepening its understanding within this context. The result of the self-attention stage is a richer, context-sensitive representation of each token, informed by its relationships with other tokens in the sentence. The output from self-attention is then passed through a feed-forward network (FFN), a layer that enhances the understanding of each token by applying non-linear transformations [33]. The FFN enhances each token's representation by adding layers of complexity, allowing identical words to reflect

varied meanings tailored to their specific context. After passing through the FFN, each token attains a comprehensive representation that synthesizes its foundational features, contextual attention, and non-linear transformations, optimizing it for advanced NLP tasks such as classification and information extraction. Through this entire process, BERT is able to comprehend text at a deeper level, capturing meaning not only at the individual word level but also through the intricate relationships between words within a sentence.

Table 2. BERT architecture details

Layer (type)	Output shape	Param#
BertForSequenceClassification	(batch_size, sequence_length, 768)	109.5M (total)
Embedding layer (word, position, token)	(batch_size, sequence_length, 768)	23,917,824
Encoder layer (12 x BertLayer)	(batch_size, sequence_length, 768)	85,441,536
Self-attention (query, key, value)	(batch_size, sequence_length, 768)	1,771,776 (per layer)
Self-attention output (dense, norm, dropout)	(batch_size, sequence_length, 768)	1,180,032 (per layer)
Intermediate dense layer (GELU)	(batch_size, sequence_length, 3072)	2,362,368 (per layer)
Output dense layer (norm, dropout)	(batch_size, sequence_length, 768)	2,364,672 (per layer)
Pooler dense layer (Tanh)	(batch_size, 768)	590,592
Classifier layer	(batch_size, num_labels)	1,538

Table 3. BERT process representations

Tokens	Embedding (dim 768)	Self-attention output (dim 768)	FFN output (dim 768)
CLS	[0.11, 0.02, ..., 0.19]	[0.15, -0.05, ..., 0.65]	[0.18, 0.09, ..., 0.60]
cars	[0.23, -0.13, ..., 0.45]	[-0.12, 0.45, ..., 0.78]	[0.22, -0.15, ..., 0.66]
have	[0.13, -0.21, ..., 0.21]	[0.12, 0.32, ..., 0.41]	[0.20, -0.19, ..., 0.33]
been	[0.08, 0.29, ..., -0.08]	[0.17, -0.05, ..., 0.28]	[-0.04, 0.09, ..., 0.47]
around	[0.16, -0.12, ..., 0.25]	[0.23, -0.18, ..., 0.14]	[0.21, 0.11, ..., 0.64]
since	[-0.03, 0.15, ..., 0.36]	[0.08, -0.14, ..., 0.21]	[0.13, 0.06, ..., -0.09]
they	[0.27, -0.09, ..., -0.02]	[-0.04, 0.24, ..., 0.33]	[0.10, -0.14, ..., 0.68]
available	[0.13, 0.22, ..., -0.27]	[0.07, 0.19, ..., 0.53]	[0.09, -0.17, ..., 0.36]
public	[-0.05, -0.11, ..., 0.45]	[0.11, 0.08, ..., -0.14]	[0.03, 0.12, ..., 0.57]
…	…	…	…
affair	[0.09, -0.21, ..., 0.11]	[-0.11, 0.29, ..., 0.33]	[0.19, 0.02, ..., 0.62]
with	[-0.08, 0.14, ..., -0.18]	[0.15, -0.27, ..., 0.04]	[0.12, -0.07, ..., 0.43]
its	[0.22, -0.05, ..., 0.19]	[0.10, 0.15, ..., -0.03]	[0.17, -0.10, ..., 0.32]
vehicles	[0.15, 0.11, ..., -0.22]	[0.28, -0.05, ..., 0.16]	[-0.12, 0.13, ..., 0.58]
seems	[0.19, -0.03, ..., 0.11]	[0.09, -0.08, ..., 0.12]	[0.07, 0.25, ..., 0.27]
eternal	[0.05, 0.07, ..., -0.04]	[0.22, -0.02, ..., -0.14]	[-0.06, 0.16, ..., 0.32]
SEP	[0.12, 0.08, ..., 0.16]	[0.10, 0.19, ..., -0.11]	[0.14, -0.18, ..., 0.27]

3.3. Result of fine-tuning and hyperparameter optimization

An initial fine-tuning of the BERT model is conducted with default parameters to establish a reliable baseline before applying the PSO algorithm for parameter optimization. Table 4 presents the initial performance results of the BERT model without specific optimization, serving as a reference for assessing performance improvements after the parameter optimization process.

As shown in Table 4, the BERT model with baseline parameters achieves accuracy levels between 83% and 94%. Although this indicates reasonably strong performance, there remains potential for further enhancement in evaluation metrics like F1-score, precision, and recall. In NLP modeling with BERT, selecting non-ideal parameter combinations can limit performance, particularly in sophisticated tasks like detecting AI-generated essays [34]. Consequently, employing parameter optimization methods is essential to maximize the model's effectiveness. PSO is applied for hyperparameter tuning to enhance the BERT model's performance in detecting AI-generated essays.

PSO is a metaheuristic approach recognized for its effectiveness in optimizing parameters across machine learning and deep learning models, including those used in NLP. PSO was chosen due to its efficiency in exploring the parameter space and its effectiveness in uncovering optimal configurations for deep learning models [35], [36]. In our research, the optimization focuses on three main hyperparameters are learning rate, batch size, and number of epochs to minimize evaluation loss on the validation set. The search boundaries are carefully defined within an appropriate range, specifying the learning rate between 1×10^{-6} and 1×10^{-4}, batch size is constrained between 16 and 64, and epochs are set from 1 to 5 to prevent extreme configurations that could destabilize the training process. The PSO objective function, implemented via the Trainer in the Transformers library, assesses parameter combinations based on evaluation loss. Validation is performed every 500 steps, and early stopping is employed to end training if no significant improvement is observed across three consecutive evaluations. PSO is configured to operate a particle swarm of size 10 for a

maximum of 5 iterations, where each particle represents a candidate hyperparameter combination that is gradually optimized in each iteration based on its fitness score. Table 5 summarizes the outcomes of PSO-optimized hyperparameter configurations applied to the BERT model.

Table 5 shows that, overall, the tested models demonstrate excellent performance, as indicated by high values in accuracy, F1, precision, and recall. The combination of an appropriate learning rate and optimal batch size positively impacts model training and generalization. While all models perform well, slight variations in outcomes are observed across different parameter combinations. The most optimal configuration was found with a learning rate of 3.47e-6, a batch size of 47, and approximately 2.7 epochs, resulting in a training accuracy of 99.46%.

To enhance the reproducibility of our approach, we provide a clear description of the integration process between BERT and PSO. The input essays first undergo preprocessing and tokenization using the BERT tokenizer, after which they are fed into the BertForSequenceClassification model. The PSO algorithm initializes a population of particles, each representing a possible combination of hyperparameters such as learning rate, batch size, and number of epochs. During training, each particle's configuration is evaluated based on validation loss, and the particles update their positions by considering both their own best performance and the global best solution found so far. This process continues iteratively until an optimal configuration is identified. The best hyperparameter set is then used to fine-tune the BERT model for final evaluation.

Table 6 provides a direct comparison between the best-performing baseline hyperparameter setting and the configuration identified through PSO optimization. The baseline setup used traditional manual tuning, with a learning rate of 5e-5, batch size of 32, and 5 epochs, which achieved an accuracy of 93.54% and an F1-score of 93.31%. While this result is acceptable, the PSO-optimized configuration significantly outperformed it, achieving an accuracy of 99.46% and an F1-score of 99.31%. The optimized configuration used a much smaller learning rate (3.47e-6), a batch size of 47, and approximately 2.7 epochs. These values, found through PSO's intelligent search, enabled the model to generalize better and converge more effectively. Notably, the F1-score, precision, and recall also increased, indicating a better balance between true positive and false positive rates. This comparison highlights the effectiveness of PSO in finding hyperparameter combinations that are difficult to determine manually and demonstrates its value in boosting BERT's performance in AI-generated essay detection tasks.

Table 4. Fine-tuning parameter result

Learning rate	Batch size	Num epoch	Accuracy	F1-score	Precision	Recall
0.00005	32	3	0.837213	0.832154	0.829342	0.835978
0.00003	16	2	0.843789	0.839478	0.836542	0.841207
0.00004	16	4	0.888214	0.885302	0.883215	0.887406
0.00002	32	3	0.879876	0.874568	0.871215	0.877543
0.00003	32	2	0.861235	0.857634	0.854213	0.859176
0.00002	16	4	0.902541	0.899876	0.89721	0.901567
0.00005	32	5	0.935412	0.933125	0.930123	0.934789
0.00003	64	3	0.872134	0.869234	0.866789	0.870456
0.00004	32	3	0.894123	0.891567	0.889456	0.892345
0.00002	16	2	0.845876	0.841234	0.839045	0.843567

Table 5. Hyperparameter optimization using PSO

Learning rate	Batch size	Num epoch	Accuracy	F1-score	Precision	Recall
5.7146799887729164e-05	63.18600623893258	1.0820392629353033	0.990751	0.988097	0.989943	0.986258
4.5948985253369004e-05	61.36109422838466	1.6929877319361548	0.989302	0.986270	0.985424	0.987117
3.677257797075701e-05	33.449619395346865	4.9775010512734355	0.993760	0.992007	0.989183	0.994847
1.552977210036071e-05	19.395938982978734	1.2671950920649016	0.992423	0.990308	0.986091	0.994561
3.4068034050765835e-05	41.109285306354934	3.426334505152745	0.994317	0.992719	0.990034	0.995419
3.1870877408661424e-05	37.259211991839564	3.946411515670476	0.990528	0.987935	0.979730	0.996278
3.1793765829040387e-05	34.85564348269941	3.776161762096859	0.989191	0.986251	0.976698	0.995992
7.194896932060413e-05	36.14931078047945	3.88616745384387	0.985514	0.981664	0.967473	0.996278
6.648045398068937e-05	30.636692308316327	2.7186498529922023	0.992311	0.990181	0.984437	0.995992
6.286899537879016e-05	35.06607073905764	2.193754986956619	0.992868	0.990868	0.987767	0.993988
3.505986939871567e-05	64.0	3.5845555198717003	0.994540	0.992993	0.992000	0.993988
3.213082701434468e-05	64.0	3.744649743736594	0.993648	0.991861	0.989459	0.994274
2.4352049736645893e-05	55.08390610302284	5.0	0.994540	0.992991	0.992281	0.993702
3.474498205359899e-06	47.41987485228536	2.6752473132377306	0.994651	0.993133	0.992565	0.993702
2.254341612465883e-05	61.549770662228056	4.757666330477469	0.994094	0.992423	0.991148	0.993702
2.0790254216330504e-05	58.54138861691617	5.0	0.994428	0.992849	0.991998	0.993702
2.1302857951125973e-05	55.82945111943465	5.0	0.994205	0.992565	0.991431	0.993702
4.9012459265130996e-05	56.539209522921624	4.862430641394423	0.993537	0.991714	0.989735	0.993702
5.114880845816846e-05	52.33458954901701	4.014062023978737	0.991420	0.989036	0.983853	0.994274

Table 6. Baseline vs PSO-optimized hyperparameter comparison

Method	Learning rate	Batch size	Num epoch	Accuracy	F1-score	Precision	Recall
Baseline (best)	5e-5	32	5	93.54%	93.31%	93.01%	93.47%
PSO-optimized	3.47e-6	47	2.7	99.46%	99.31%	99.26%	99.37%

Figure 2 illustrates a well-aligned relationship between training loss and validation loss throughout the model's training process. The training loss decreases rapidly at the start of training, indicating that the model quickly learns patterns from the training data. After this sharp decline, the training loss remains low with minor fluctuations, likely due to batch variations during training. Similarly, the validation loss also drops swiftly early in the training and stabilizes at a low level, with only slight fluctuations. As both training and validation loss levels stabilize without any noticeable divergence, there is no indication of overfitting. The model preserves its ability to generalize to unseen data, as reflected by the parallel trends in both loss curves. This relationship indicates that the model learns effectively from the training data while maintaining its ability to perform well on validation data.

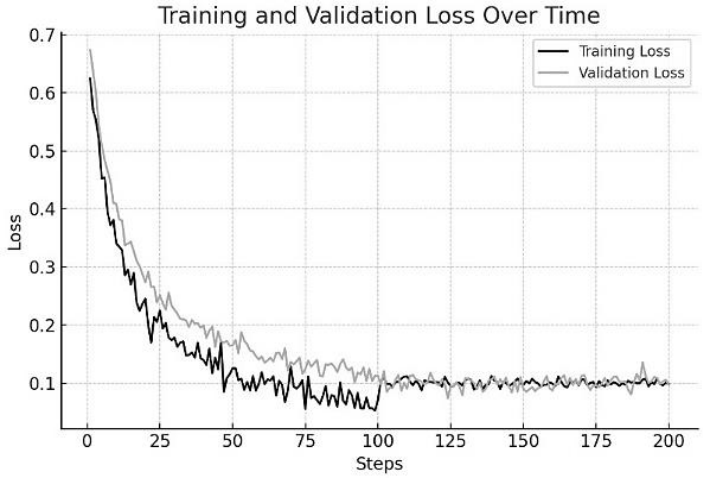


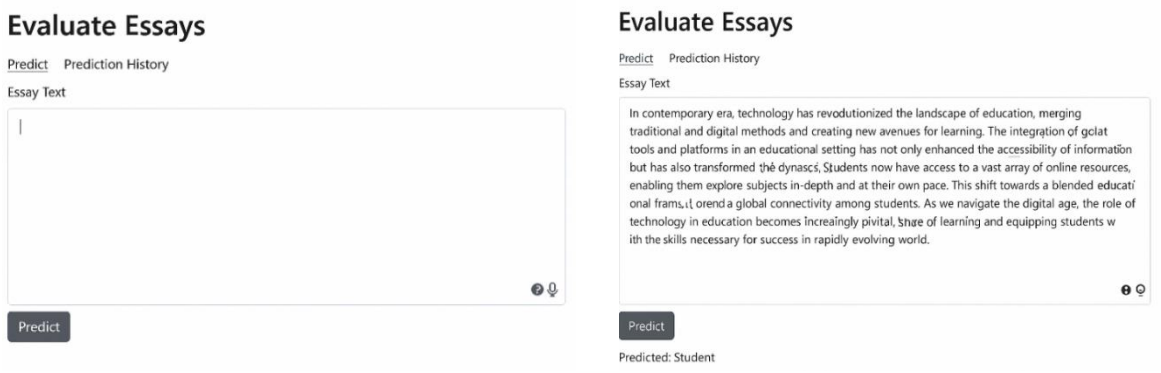
Figure 2. Training and validation loss graph

3.4. Result of evaluation

In pursuit of higher detection accuracy, this study not only focuses on developing an effective model but also integrates it into a web-based interface. This deployment allows users to perform real-time detection of AI-generated essays, as demonstrated in Figure 3, enhancing both usability and accessibility for practical applications. The deployment of the optimized BERT model for detecting AI-generated essays starts by integrating the model into an API built with FastAPI. The API receives essay text from users, processes it, and generates a prediction on whether the text was authored by a human or produced by AI. A user-friendly web interface was created, including a text input field and a "Predict" button that sends the text to the API and displays the outcome. After deployment, this web application allows users to input essay text and instantly receive detection results, determining if the essay was likely generated by AI, as shown in Figure 3(a). In Figure 3(b), a user submits an essay for analysis, which the application processes, returning a "Student" label suggesting that the text was most likely written by a human.

The deployed model enables thorough testing on a dataset of 9,250 essays, predicting whether each text was generated by AI or written by a human. In addition to accuracy-based evaluation, we also measured system performance in terms of response time and load testing. During real-time prediction via the FastAPI-based interface, the average response time was recorded at 312 milliseconds per request, with a maximum observed latency of 487 milliseconds under normal conditions. The system was further tested using simulated concurrent user requests (load test with 500 requests per minute), which showed stable performance with 98.2% of requests completed under 500 ms, indicating strong scalability for moderate traffic scenarios. Furthermore, informal user feedback was collected from a group of 20 students and lecturers who used the system for evaluating essay content. The majority (85%) reported that the system was intuitive and helpful, especially in quickly assessing the authenticity of academic writing. Several users appreciated the clarity of the prediction result, though a few suggested the addition of more detailed

explanations alongside the binary classification outcome. These metrics and feedback reinforce the system's practicality not only in terms of detection accuracy but also in user experience and operational efficiency under real-time usage conditions.



(a) (b)

Figure 3. Web interface essay-AI detection (a) input text and (b) result detection

The predictions are then compared to the actual labels to assess accuracy and additional performance metrics, providing a clear evaluation of the model's effectiveness. The evaluation employs key performance metrics, including accuracy, precision, recall, and F1-score, to rigorously assess the model's effectiveness in differentiating AI-generated essays from human-authored texts. It further examines the application's real-time operational efficiency, analyzing response times and reliability across varied essay types to ensure dependable and consistent performance. The evaluation results of the model after testing on 9,250 essays are presented in the confusion matrix shown in Table 7.

Table 7. Confussion metrix result

Predicted/actual	Human-written (0)	AI-generate (1)
Human-written (0)	4750	100
AI-generate (1)	50	4350

Table 7 highlights the model's strong proficiency in distinguishing AI-generated essays from those written by humans. With an impressive 98% accuracy, the model consistently classifies nearly all essays correctly, demonstrating high reliability across various cases. The precision of 97.75% indicates the model's accuracy in identifying AI-generated content with minimal misclassification of human-authored texts, reflecting its selectiveness and precision. Additionally, a recall of 98.87% underscores the model's ability to capture almost all AI-generated essays, minimizing undetected instances. The F1-Score of 98.31% further illustrates a well-balanced performance between precision and recall, confirming the model's robustness in accurate detection while minimizing errors. These metrics collectively affirm the model's high accuracy, dependability, and suitability for real-world applications in AI-generated essay detection.

4. CONCLUSION

This research presents an innovative approach for detecting AI-generated essays by integrating the BERT model, enhanced through PSO. BERT was selected due to its superior ability to interpret complex language nuances, although fine-tuning its hyperparameters continues to be a significant challenge. PSO, inspired by the collective behavior of bird flocks, is applied to automate the optimization of key parameters, including learning rate, batch size, and number of epochs. This method reduces reliance on time-consuming manual tuning, streamlining the process and enhancing efficiency in model training. The initial evaluation of the BERT model without optimization showed an accuracy ranging from 83% to 94%. After applying PSO and testing the model with a validation dataset post-deployment, it maintained high accuracy of up to 98%, with an F1-Score of 98.31%, precision of 97.75%, and recall of 98.87%, indicating a significant improvement in performance. The test dataset for this research comprised 9,250 essays, and the results demonstrated that the model effectively distinguished between AI-generated and human-written essays. This

was evident from the confusion matrix, which showed that 4,750 human-written essays and 4,350 AI-generated essays were correctly classified. The implementation through a web interface based on FastAPI allows for real-time essay detection, making it easier for users to verify text authenticity. This interface provides a text input field and a "Predict" button to generate instant predictions. This research confirms that the PSO-optimized BERT approach not only improves the accuracy and efficiency of detection but also plays a vital role in upholding academic integrity. The system offers substantial benefits to the educational field, where AI-generated academic content is becoming more prevalent. Additionally, this method has the potential for further development in future NLP applications, aiding in the identification of AI-generated material and ensuring the quality and authenticity of written content in the digital age. The novelty of this work lies in the integration of PSO to systematically optimize the hyperparameters of the BERT model, a method not widely explored for AI-generated content detection in academic contexts. In addition to improving performance, this approach enables practical real-time deployment, with implications for broader use in detecting synthetic content in other sectors such as digital media, regulatory compliance, and content authenticity verification.

ACKNOWLEDGEMENTS

This work was supported in part by LPPM Institut Teknologi Del, Indonesia through funding No. 021.35/ITDel/LPPM/Penelitian/IV/2024.

REFERENCES

[1] N. Tyagi and B. Bhushan, "Demystifying the role of natural language processing (NLP) in smart city applications: background, motivation, recent advances, and future research directions," *Wireless Personal Communications*, vol. 130, no. 2, pp. 857–908, 2023.

[2] O. Ali, P. A. Murray, M. Momin, Y. K. Dwivedi, and T. Malik, "The effects of artificial intelligence applications in educational settings: Challenges and strategies," *Technological Forecasting and Social Change*, vol. 199, p. 123076, Feb. 2024, doi: 10.1016/j.techfore.2023.123076.

[3] F. Kamalov, D. Santandreu Calonge, and I. Gurrib, "New era of artificial intelligence in education: towards a sustainable multifaceted revolution," *Sustainability*, vol. 15, no. 16, p. 12451, Aug. 2023, doi: 10.3390/su151612451.

[4] P. P. Ray, "ChatGPT:a comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope," *Internet of Things and Cyber-Physical Systems*, vol. 3, pp. 121–154, 2023, doi: 10.1016/j.iotcps.2023.04.003.

[5] S. Grassini, "Shaping the future of education: exploring the potential and consequences of AI and ChatGPT in educational settings," *Education Sciences*, vol. 13, no. 7, p. 692, Jul. 2023, doi: 10.3390/educsci13070692.

[6] S. Jamshidi *et al.*, "Effective text classification using BERT, MTM LSTM, and DT," *Data & Knowledge Engineering*, vol. 151, p. 102306, May 2024, doi: 10.1016/j.datak.2024.102306.

[7] Y. Mao, Q. Liu, and Y. Zhang, "Sentiment analysis methods, applications, and challenges: a systematic literature review," *Journal of King Saud University - Computer and Information Sciences*, vol. 36, no. 4, p. 102048, Apr. 2024, doi: 10.1016/j.jksuci.2024.102048.

[8] J. Campino, "Unleashing the transformers: NLP models detect AI writing in education," *Journal of Computers in Education*, vol. 12, no. 2, pp. 645–673, 2025.

[9] M. P. Geetha and D. Karthika Renuka, "Improving the performance of aspect based sentiment analysis using fine-tuned BERT base uncased model," *International Journal of Intelligent Networks*, vol. 2, pp. 64–69, 2021, doi: 10.1016/j.ijin.2021.06.005.

[10] P. Dhiman, A. Kaur, D. Gupta, S. Juneja, A. Nauman, and G. Muhammad, "GBERT: a hybrid deep learning model based on GPT-BERT for fake news detection," *Heliyon*, vol. 10, no. 16, p. e35865, Aug. 2024, doi: 10.1016/j.heliyon.2024.e35865.

[11] A. K. Pandey and S. S. Roy, "CANBLWO: a novel hybrid approach for semantic text generation," *The International Arab Journal of Information Technology*, vol. 21, no. 4, 2024, doi: 10.34028/iajit/21/4/11.

[12] L. Zhao, W. Gao, and J. Fang, "Optimizing large language models on multi-core CPUs: a case study of the BERT model," *Applied Sciences*, vol. 14, no. 6, p. 2364, Mar. 2024, doi: 10.3390/app14062364.

[13] Y. Zhong and S. D. Goodfellow, "Domain-specific language models pre-trained on construction management systems corpora," *Automation in Construction*, vol. 160, p. 105316, Apr. 2024, doi: 10.1016/j.autcon.2024.105316.

[14] C. Maraveas, P. G. Asteris, K. G. Arvanitis, T. Bartzanas, and D. Loukatos, "Application of bio and nature-inspired algorithms in agricultural engineering," *Archives of Computational Methods in Engineering*, vol. 30, no. 3, pp. 1979–2012, Apr. 2023, doi: 10.1007/s11831-022-09857-x.

[15] L. Bayuaji, Kusnadi, M. Y. Amzah, and D. Pebrianti, "Optimization of feature selection in support vector machines (SVM) using recursive feature elimination (RFE) and particle swarm optimization (PSO) for heart disease detection," in *2024 9th International Conference on Mechatronics Engineering (ICOM)*, IEEE, Aug. 2024, pp. 304–309. doi: 10.1109/ICOM61675.2024.10652561.

[16] W. Wiharto, Y. Mufidah, U. Salamah, E. Suryani, and S. Setyawan, "The use of genetic algorithm and particle swarm optimization on tiered feature selection method in machine learning-based coronary heart disease diagnosis system," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 14, no. 4, pp. 4563–4576, Aug. 2024, doi: 10.11591/ijece.v14i4.pp4563-4576.

[17] S. A. Bin-Nashwan, M. Sadallah, and M. Bouteraa, "Use of ChatGPT in academia: academic integrity hangs in the balance," *Technology in Society*, vol. 75, p. 102370, Nov. 2023, doi: 10.1016/j.techsoc.2023.102370.

[18] R. F. Kizilcec *et al.*, "Perceived impact of generative AI on assessments: comparing educator and student perspectives in Australia, Cyprus, and the United States," *Computers and Education: Artificial Intelligence*, vol. 7, p. 100269, Dec. 2024, doi: 10.1016/j.caeai.2024.100269.

[19] C. Zhai, S. Wibowo, and L. D. Li, "The effects of over-reliance on AI dialogue systems on students' cognitive abilities: a systematic review," *Smart Learning Environments*, vol. 11, no. 1, p. 28, Jun. 2024, doi: 10.1186/s40561-024-00316-7.

[20] H. Johnston, R. F. Wells, E. M. Shanks, T. Boey, and B. N. Parsons, "Student perspectives on the use of generative artificial intelligence technologies in higher education," *International Journal for Educational Integrity*, vol. 20, no. 1, 2024, doi: 10.1007/s40979-024-00149-4.
[21] Supriyono, A. P. Wibawa, Suyono, and F. Kurniawan, "Advancements in natural language processing: implications, challenges, and future directions," *Telematics and Informatics Reports*, vol. 16, p. 100173, Dec. 2024, doi: 10.1016/j.teler.2024.100173.
[22] F. I. Kurniadi, N. L. P. S. P. Paramita, E. F. A. Sihotang, M. S. Anggreainy, and R. Zhang, "BERT and RoBERTa models for enhanced detection of depression in social media text," *Procedia Computer Science*, vol. 245, pp. 202–209, 2024, doi: 10.1016/j.procs.2024.10.244.
[23] B. Min *et al.*, "Recent advances in natural language processing via large pre-trained language models: a survey," *ACM Computing Surveys*, vol. 56, no. 2, pp. 1–40, Feb. 2024, doi: 10.1145/3605943.
[24] I. A. Zahid *et al.*, "Unmasking large language models by means of openAI GPT-4 and Google AI: a deep instruction-based analysis," *Intelligent Systems with Applications*, vol. 23, p. 200431, Sep. 2024, doi: 10.1016/j.iswa.2024.200431.
[25] L. Abualigah, "Particle swarm optimization: advances, applications, and experimental insights," *Computers, Materials & Continua*, vol. 82, no. 2, pp. 1539–1592, 2025, doi: 10.32604/cmc.2025.060765.
[26] T. A. Prasetyo, "Particle swarm optimization and genetic algorithm for big vehicle problem: case study in national pure milk company," *International Journal of Computing Science and Applied Mathematics*, vol. 7, no. 1, p. 28, Feb. 2021, doi: 10.12962/j24775401.v7i1.8210.
[27] T. A. Prasetyo, V. L. Desrony, H. F. Panjaitan, R. Sianipar, and Y. Pratama, "Corn plant disease classification based on leaf using residual networks-9 architecture," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 13, no. 3, pp. 2908–2920, Jun. 2023, doi: 10.11591/ijece.v13i3.pp2908-2920.
[28] T. A. Prasetyo *et al.*, "Evaluating the efficacy of univariate LSTM approach for COVID-19 data prediction in Indonesia," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 34, no. 2, pp. 1353–1366, May 2024, doi: 10.11591/ijeecs.v34.i2.pp1353-1366.
[29] A. Subakti, H. Murfi, and N. Hariadi, "The performance of BERT as data representation of text clustering," *Journal of Big Data*, vol. 9, no. 1, p. 15, Dec. 2022, doi: 10.1186/s40537-022-00564-9.
[30] A. Mumuni and F. Mumuni, "Automated data processing and feature engineering for deep learning and big data applications: A survey," *Journal of Information and Intelligence*, vol. 3, no. 2, pp. 113–153, Mar. 2025, doi: 10.1016/j.jiixd.2024.01.002.
[31] Y. Ortakci, "Revolutionary text clustering: investigating transfer learning capacity of SBERT models through pooling techniques," *Engineering Science and Technology, an International Journal*, vol. 55, p. 101730, Jul. 2024, doi: 10.1016/j.jestch.2024.101730.
[32] D. S. Asudani, N. K. Nagwani, and P. Singh, "Impact of word embedding models on text analytics in deep learning environment: a review," *Artificial Intelligence Review*, vol. 56, no. 9, pp. 10345–10425, Sep. 2023, doi: 10.1007/s10462-023-10419-1.
[33] E. Y. Zhang, A. D. Cheok, Z. Pan, J. Cai, and Y. Yan, "From turing to transformers: a comprehensive review and tutorial on the evolution and applications of generative transformer models," *Sci*, vol. 5, no. 4, p. 46, Dec. 2023, doi: 10.3390/sci5040046.
[34] X. Yang *et al.*, "FinChain-BERT: a high-accuracy automatic fraud detection model based on NLP methods for financial scenarios," *Information*, vol. 14, no. 9, p. 499, Sep. 2023, doi: 10.3390/info14090499.
[35] A. G. Gad, "Particle swarm optimization algorithm and its applications: a systematic review," *Archives of Computational Methods in Engineering*, vol. 29, no. 5, pp. 2531–2561, Aug. 2022, doi: 10.1007/s11831-021-09694-4.
[36] A. Raponi and D. Marchisio, "Deep learning for kinetics parameters identification: a novel approach for multi-variate optimization," *Chemical Engineering Journal*, vol. 489, p. 151149, Jun. 2024, doi: 10.1016/j.cej.2024.151149.

BIOGRAPHIES OF AUTHORS

Tegar Arifin Prasetyo is a current lecturer and researcher member in the Information Technology Department at Institut Teknologi Del since 2020. He has experience specializing in building mathematical models, machine learning models, analytical Android tools development, control systems, and computer programming. He dedicates himself to university teaching and conducting research. His research interests include artificial intelligence, machine learning and algorithm computation, optimal control, and mathematical modeling in epidemiology and bioinformatics. He can be contacted at email: tegar.prasetyo@del.ac.id or arifintegar12@gmail.com.

Rudy Chandra is a current lecturer and researcher member in the Information Technology Department at Institut Teknologi Del since 2022. He has experience specializing in building machine learning models, artificial neural networks, decision support systems, distributed systems, software testing, and computer programming. He dedicates himself to university teaching and conducting research. His research interests include artificial intelligence, neural networks, machine learning and algorithm computation, decision support systems, distributed systems, and software testing. He can be contacted at email: rudy.chandra@del.ac.id or rudychandra0@gmail.com.

Wesly Mailander Siagian is a current lecturer and researcher member in the Engineering Management Department at Institut Teknologi Del since 2020. As a professional trader, he must be detail-oriented and know exactly when the best time to trade is. He always follows risk and money management rules and never lets one losing trade wipe out a significant amount of his capital. His research interests include machine learning and algorithm computation in forecasting models. He can be contacted at email: wesly.siagian@del.ac.id.

Horas Marolop Amsal Siregar is an alumnus of the Information Technology Department at Institut Teknologi Del. He also participated in the Huawei Summer Camp in China in 2022, focusing on AI, and won first place in the Indonesia region for the Huawei ICT event with the best innovation track. He has experience in artificial intelligence modeling and software engineering for web and mobile applications. His research interests include artificial intelligence and machine learning methods applied to web infrastructure and mobile programming. He can be contacted at email: if321051@students.del.ac.id.

Samuel Jefri Saputra Siahaan is an alumnus of the Information Technology Department at Institut Teknologi Del. He has experience in sales forecasting using several methods, web development, and computer programming. His research interest includes sales forecasting methods. He can be contacted at email: if321037@students.del.ac.id.