J-COSINE (Journal of Computer Science and Informatics Engineering)
Vol. 9, No. 2, December 2025 E-ISSN:2541-0806
Accredited Sinta-4 by RISTEKDIKTI Decree No. 79/E/KPT/2023 P-ISSN:2540-8895

Comparative Analysis of Proposed CNN Performance with CNN and Naive Bayes from Kaggle in ChatGPT Tweet Sentiment Analysis

Alwi Pratama*, Ario Yudo Husodo, Fitri Bimantoro

Dept. Informatics Engineering, University of University
Jl. Majapahit 62, Mataram, Lombok NTB, INDONESIA
Email: alwigra2@gmail.com, ario@unram.ac.id, bimo@unram.ac.id

Corresponding Author

Abstract **The increasing use of social media such as Twitter has increased the demand for efficient sentiment analysis. This study compares the performance of the author's CNN model with Kaggle's CNN and Naïve Bayes in analyzing ChatGPT-related tweets, focusing on accuracy, training time, and inference speed. The author's CNN achieves an accuracy of 89%, higher than Kaggle's CNN (85%) but slightly lower than Naïve Bayes (90%), which excels in recognizing word patterns through its probabilistic approach. However, the author's CNN shows the fastest inference time (3.0563 seconds) compared to Kaggle's CNN (5.612 seconds) and Naïve Bayes (4.0366 seconds). This efficiency is due to the use of GlobalMaxPooling1D which reduces data size and computational load, and processes batches of 128 data at once, making it more efficient in running predictions than if done one by one. In, Kaggle CNN training is much slower (1309.5718 seconds) compared to the author's CNN (148.9196 seconds), while Naïve Bayes is the fastest (0.1029 seconds). The training speed that makes the Naïve Bayes model superior is because it only calculates probabilities based on feature distributions without requiring iterations like backpropagation in CNN. This study highlights the trade-off between accuracy and efficiency in sentiment analysis models.**

***Keywords*: Sentiment analysis, CNN, Naive Bayes, Inference time, performance evaluation.**

I. Introduction

Sentiment analysis refers to a set of techniques and approaches used to detect and extract subjective information, such as opinions, feelings, or attitudes from text data. This process is often used to assess the emotional tone or sentiment contained in a text, allowing the classification of opinions into positive and negative or neutral. In addition, sentiment plays an important role in evaluating the perception of public performance, social media discussions, and certain services. By analyzing these sentiments, we can make decisions and improve communication and development strategi. [1]

Social media platforms are now very popular in the digital era to convey views, opinions, and feelings widely. One of the largest social networks, Twitter, is used by millions of people every day to share their thoughts, including about ChatGPT. ChatGPT itself is a text-based artificial intelligence developed by OpenAI, allowing more natural interactions with humans and supporting various types of requests. Therefore, sentiment analysis is important to understand how Twitter users assess the performance and public acceptance of this AI function.[2]

X or Twitter is the subject of sentiment analysis research because of the abundance of data available on the platform. With its wide reach, X allows researchers to observe current trends and spread information quickly. In addition, X provides access to its data, making it easier to collect and analyze data. The available data is in the form of user uploads in the form of Indonesian language text, which can contain positive, neutral, or negative opinions. To understand the feelings contained in the opinion, an evaluation and assessment of the text data is carried out, which ultimately produces valuable information. This process in text processing is known as sentiment analysis.[3]

The dataset used in this study is a collection of tweets from Kaggle collected over a month, containing public responses regarding various topics and sentiments surrounding ChatGPT. This dataset includes various types of sentiments, ranging from positive, negative, and neutral. which reflects the diverse reactions of Twitter users to the development, use, and issues related to ChatGPT technology[4]. During the collection, these tweets not only included personal comments or opinions, but also discussions about various aspects related to ChatGPT. including technical capabilities, potential applications, ethical issues and even problems related to bias or misinformation that may be caused by the technology. Therefore, this dataset is very rich in various types of sentiments that need to be analyzed to get a complete picture of public perception of ChatGPT.

In previous studies that have been studied, it was said there that the Naive Bayes method is very good in terms of sentiment analysis because the Naive Bayes method will study the characteristics of the words in each class. while on the one hand the use of the CNN method which is part of the Artificial Neural Network (ANN) is able to achieve good performance in sentence classification, and also the use of this CNN method can also detect sarcastic on twitter[5]. The comparison of CNN and Naïve Bayes is done because both have different approaches in sentiment analysis, CNN captures complex patterns in text but is slower, while Naïve Bayes is faster and more efficient but lacks understanding of context. This analysis helps determine the trade-off between accuracy, speed, and efficiency for real-time applications.

Based on the literature study that has been conducted, there are still few that examine the number of epochs applied in the CNN model. Then in previous studies there has been no exploration of the efficiency of the CNN model in tweet sentiment analysis, especially in terms of the balance between accuracy, training time, and inference time. Most previous studies have focused more on improving model accuracy without paying attention to the balance of testing rather than processing the dataset[6]. On the other hand, direct comparisons between self-developed CNNs and models available on platforms such as Kaggle are still rare, especially in ChatGPT sentiment analysis. Therefore, this study fills the gap by comparing self-developed CNN models with CNN models from Kaggle and Naive Bayes from Kaggle, in order to provide insight into the trade-offs between accuracy, training time, and inference time.

One aspect of evaluation that can be used to assess and develop ChatGPT is through sentiment analysis of its users. This analysis helps understand how models like ChatGPT are perceived by the public.[7] The results of this analysis can be used to improve and strengthen model performance. This study conducted a sentiment analysis of platform X users, previously known as Twitter, using the CNN model developed by the author. The model successfully achieved a higher level of accuracy than the CNN model from Kaggle. In addition, the CNN model designed by the author also showed a faster inference time compared to the Naive Bayes model taken from Kaggle.

II. Literature Review

Artificial intelligence has made a significant impact on various aspects of human life, becoming a key element in work, social interactions, and decision-making. This technology able to understand, learn, and make decisions based on available data.[3] Over time, AI algorithms have continued to advance to be able to process data in its original form, allowing for analysis of unstructured data such as raw text and images. AI algorithms, such as deep learning, have made rapid progress. Models such as Convoluted Neutral Networks are now increasingly recognized for their ability to capture local patterns in text such as phrases and n-grams.

The use of chatbots is increasingly popular among global netizens over time. Chatbots have the ability support various human activities, both in the world of work and everyday life. Currently, chatbots have become more sophisticated with the support of artificial intelligence, making them the main choice for many users to complete various daily tasks. ChatGPT has emerged as one of the most sophisticated chatbots since early 2018 and has continued to be popular, especially since 2020. By utilizing Natural Language Processing (NLP) technology, ChatGPT able to understand and meet user needs effectively.[8]

Twitter or what we now know as X has become an object of research in sentiment analysis because of the availability of abundant data in it. With a fairly reach, X allows researchers to study current trends while disseminating information quickly. In addition, X provides data access that facilitates the collection and analysis process for researchers. The available data is in the form of user uploads in the form of Indonesian and foreign language texts, these texts contain opinions with positive, neutral, and negative sentiments. To understand the feelings contained in the opinion, an analysis is carried out that evaluates and assesses textual data, producing valuable information. This process is known as text processing as sentiment analysis. [2]

ChatGPT sentiment analysis was performed using the CNN method created by the author and the CNN method and the naive Bayes method taken from Kaggle. This algorithm is used to compare the accuracy, training time and inference time of each model in analyzing the sentiment of the tweet dataset. The CNN method created by the author is specifically designed to capture complex patterns in text data through use of optimized convolution layers, while the CNN and Naive Bayes methods from Kaggle serve as a comparison to evaluate the performance of the author's model.

During the experiment, each algorithm was tested on the same dataset to ensure fairness in the evaluation. The CNN created by the authors showed the ability to recognize contextual relationships between words better, producing competitive accuracy compared to the method from Kaggle. However, this model requires a longer training time due to the complexity of its architecture. In contrast, the CNN from Kaggle has almost the same performance in terms of accuracy, but shows better efficiency in terms of training time due to its more generic architectural design. [9]

The CNN created by the author shows superior ability in recognizing contextual relationships between words, produces competitive accuracy and even has a faster inference time compared to the Naive Bayes model from Kaggle, although it requires a longer training time. Conversely, Naive Bayes from Kaggle, although it has a very fast training time, shows shortcomings in capturing more complex patterns, so its inference time is slower compared to the CNN created by the author.[10]

The results of this study show that the author's CNN method able to provide a balance between higher accuracy and competitive inference time, although it requires a longer training time. Meanwhile, Naive Bayes remains an efficient choice for fast analysis with smaller or simpler datasets. This analysis provides deep insights into how various algorithms can be applied to sentiment analysis tasks and shows the potential for performance improvements through more focused and tailored model design.

III. Research Methodology

The research framework in this paper will be presented in the following framework.

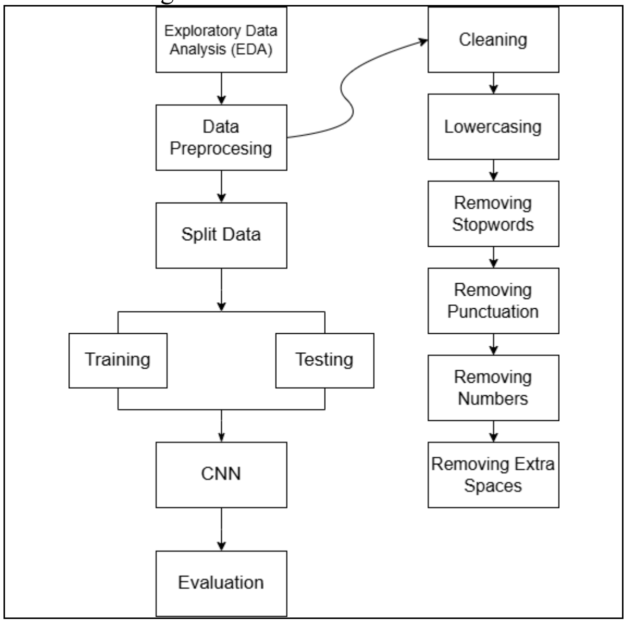
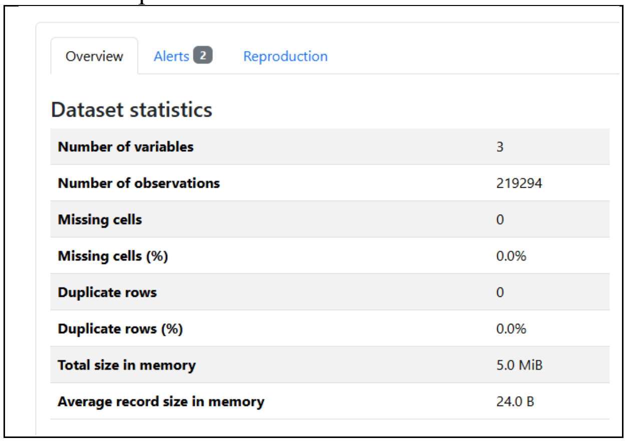


Fig 1. Research Framework flow

A. Exploratory Data Analysis (EDA)

Exploratory Data Analysis is the process of initial exploration of a dataset to understand the structure, patterns and characteristics of the data before carrying out further analysis. This step includes checking descriptive statistics, data visualization, outlier detection, identification of missing values, and analysis of relationships between variables[11]. Exploratory Data Analysis aims to find initial insights, ensure data quality, and help determine appropriate analysis or modeling strategies. Usually, tools such as Pandas, Matplotlib, and Seaborn in Python are used to make this process easier.



Dataset statistics

Number of variables	3
Number of observations	219294
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	5.0 MiB
Average record size in memory	24.0 B

Fig. 2 Data Statistics

Examples in this data there are three sentiments, namely good, neutral, and bad, in the context of user X's response to ChatGPT.

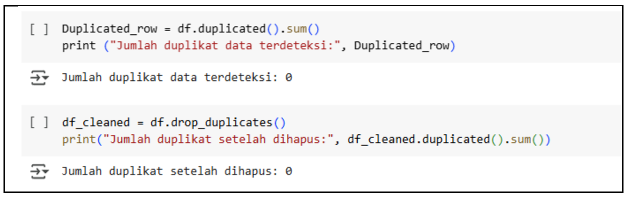
TABLE I. Review Examples of Each Sentiment

Tweets	Label
ChatGPT: Optimizing Language Models for Dialogue	Neutral
Try talking with ChatGPT, our new AI system which is optimized for dialogue. Your feedback will help us improve it	Good
Babe wake up ChatGPT just dropped	Bad

Good sentiment reflects user satisfaction, neutral sentiment indicates a response that does not indicate satisfaction or dissatisfaction clearly, and bad sentiment describes user dissatisfaction. The following are examples of each sentiment from this data as follows.

B. Data Preprocessing

The data preprocessing stage is a series of steps to clean and analyze text data so that it is ready for further use [7]. In this study, the data processed is data from collected tweets, the amount of raw data that will be processed at this stage is 219.293. This process is as shown in Figure 1 which explains that the preprocessing process carried out begins with cleaning to remove irrelevant elements such as HTML tags, special characters, or other analysis symbols that do not contribute to. Furthermore, the letters in the text are changed to lowercase to ensure consistency and avoid differences between capital and lowercase letters. Common words that have no analytical value, such as "and", "di", "or" which ", are removed at the stopword removal stage. The text is then cleaned of punctuation and numbers, unless the numbers have an important context in the analysis. Finally, excess spaces are removed to produce more structured and clean text. This process is designed to improve the quality of text data so that it can provide more accurate results in applications such as sentiment.



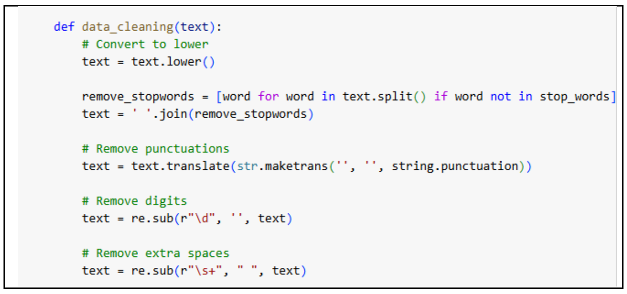
```
[ ] Duplicated_row = df.duplicated().sum()
    print ("Jumlah duplikat data terdeteksi:", Duplicated_row)

    Jumlah duplikat data terdeteksi: 0

[ ] df_cleaned = df.drop_duplicates()
    print("Jumlah duplikat setelah dihapus:", df_cleaned.duplicated().sum())

    Jumlah duplikat setelah dihapus: 0
```

Fig. 3 Cleaning Duplicate



```
def data_cleaning(text):
    # Convert to lower
    text = text.lower()

    remove_stopwords = [word for word in text.split() if word not in stop_words]
    text = ' '.join(remove_stopwords)

    # Remove punctuations
    text = text.translate(str.maketrans('', '', string.punctuation))

    # Remove digits
    text = re.sub(r"\d", '', text)

    # Remove extra spaces
    text = re.sub(r"\s+", " ", text)
```

Fig. 4 Data Preprocessing

The data preprocessing process is carried out to clean the text from irrelevant elements or elements that interfere with sentiment analysis, such as capital letters, punctuation, stop words, URLs, and excess spaces[12]. This program is more focused on processing standard texts that generally have a more formal and consistent structure, without handling the emoji, hashtag, and slang sections because they may require additional processing techniques such as slang dictionaries, emoji pattern recognition, or meaning extraction from hashtags and they do not provide significant contributions to the analysis, especially since the author's main focus is on formal linguistic patterns in the text. Table II below shows the transformation of each preprocessing stage, starting from the initial text to the final text which is cleaner and ready for further analysis.

TABLE II. RESULT TRANSFORMATION

Transformation	Before	After
Lowercase	ChatGPT: Optimizing Language Models For Dialogue (@openai)	chatGPT: optimizing language models for dialogue (@openai)
Stopwords removed	ChatGPT is optimizing language models for dialogue with OpenAI research.	ChatGPT optimizing language models dialogue OpenAI research.
Punctuation Removed	ChatGPT: optimizing language models for dialogue (@OpenAI).	ChatGPT optimizing language models for dialogue OpenAI
URL Removed	chatGPT: optimizing language models for dialogue (openai) https://t.co/2Vy57UeczH	chatGPT: optimizing language models for dialogue openai
Numbers Deleted	123 chatGPT: optimizing language models for dialogue openai	chatGPT: optimizing language models for dialogue openai
Excessive Spaces Removed	chatGPT: optimizing language models for dialogue openai	chatGPT: optimizing language models for dialogue openai

C. Split Dataset

At this stage, the dataset is divided into training data (training set) and testing data (testing set) using the training tests split function. The initial dataset is separated into features (input) and targets (output). Where 80% of the data is used for model training and 20% for testing (test size = 0.2). The proportion of 80:20 was chosen because it is a common standard that provides a sufficient balance of data for training the model and data that is capable of evaluation. Before the division, the data is shuffled (shuffle = true) to ensure an even distribution, and the use of random state = 42 ensures that the results of the data division remain consistent when the code is re-run. The result of this process is four subsets: target features for testing (in test and out test). This division is important to train the model with specific data and to emit its performance objectively on data that has never been seen, thus providing an accurate performance evaluation[13]. In the three models presented in this paper, both the author's CNN, CNN from Kaggle, and CNN from Naïve Bayes, do not use stratified splitting in dividing the training and testing data. The author realizes that without stratified splitting, there is a risk of uneven class distribution between training and testing data, especially if the dataset has a class imbalance. This can cause the model to be trained more often on the majority class and less able to recognize patterns from the minority class, resulting in bias in predictions.

D. *Convolutional Neural Network (CNN)*

In the sentiment analysis designed by the authors, Convolutional Neural Network (CNN) is used to extract local features from text data by identifying patterns that indicate sentiment starting with a 128-dimensional embedding layer to capture semantic relationships between words. The model includes a 1D convolutional relationship with 200 filters and a kernel size of 3, complemented by a ReLU activation function to enhance feature extraction and batch normalization to stabilize the activation distribution. Next, a max polling layer with a pool size of 2 is applied to reduce the feature dimensionality, followed by dropout to prevent overfitting, before the results are flattened into a 1D vector. The model has two fully connected layers, namely the first layer with 200 neurons and a ReLU activation function to capture complex patterns, and a layer with 3 output neurons and softmax activation to classify into positive, neutral, and negative categories.[6]

The model was compiled using Adam optimization with a default learning rate of 0.001, as applied automatically in Keras if not explicitly specified. Adam was chosen because of its ability to adjust the learning rate adaptively, which helps accelerate convergence and improve the stability of model training. The loss function used is sparse categorical crossentropy, which is suitable for multi-class classification problems with integer labels. In addition, this model implements early stopping to stop training if there is no improvement in the validation loss, thus avoiding overfitting and making this architecture more efficient in classification for sentiment analysis. If needed, the learning rate parameter can be further adjusted to optimize the model performance.

The CNN model from Kaggle is designed with a simple architecture consisting of Embedding layers, Conv1D, MaxPooling1D, Flatten, and Dense layers. This model uses Adam optimizer and sparse_categorical_crossentropy as the loss function to handle three-class classification. To improve training efficiency and prevent overfitting, this model applies EarlyStopping, which stops training if the val_loss does not improve after 3 epochs. This model is trained with a batch_size of 1024 and a validation split of 0.1, which means 10% of the data is used for validation.

This model is one of the baselines that are compared with other CNN models in the sentiment analysis study.

The CNN model created by the author has several key differences compared to the CNN model from Kaggle. The author's model uses a more complex architecture with 200 convolutional filters, batch normalization for stabilization, and dropout to prevent overfitting, while the Kaggle model only uses 32 filters and has no batch normalization or dropout. In addition, the author's model has two fully connected layers with 200 neurons in the first layer for deeper feature extraction, while the Kaggle model only has one dense layer with 64 neurons. Although both models use Adam optimization and sparse categorical cross-entropy loss function, the author's model is designed to be more stable and able to generalize better in sentiment analysis.

E. Evaluation

This stage is an evaluation conducted after the implementation of the Convolutional Neural Network method. This evaluation process includes three main steps.

1. Confusion Matrix

The confusion matrix section of this paper shows the performance of the model in classifying three sentiment classes, namely bad, good, and neutral. For the actual bad label (13,768 data), the model correctly predicted 12,721, but incorrectly predicted 98 as good and 949 as neutral. For the good label (11,244 data), the model correctly predicted 9,987, but incorrectly predicted 157 as bad and 1,100 as neutral. While for the neutral label (11,066 data), the model correctly predicted 9,432, but incorrectly predicted 848 as bad and 786 as good. Overall, the model showed good performance, but there were significant errors in the good class, especially those predicted as neutral.[14]

2. Accuracy

The model accuracy of 89% was obtained because the dataset looked quite balanced with almost the same amount of data in each class (13,768 for bad, 11,244 for good, and 11,066 for neutral), so that the model could learn consistently in all classes. In addition, the performance of metrics such as precision, recall, and F1-score which were quite high (82% - 93%) indicated good data processing, effective model algorithms, and representative data retribution had helped the model generalize well. However, quite significant errors occurred in the good class, where the recall precision was slightly lower (82% and 89%), possibly due to overlapping features with the neutral class or the number of features that were less discriminatory to distinguish the two classes. This shows room for improvement, such as further feature analysis or fine-tuning the model to reduce classification errors between classes. [15]

3. Classification Report

Classification report is a summary of the performance evaluation metrics of a classification model, including precision, recall, F1-score, and support for each class. Precession measures the accuracy of positive predictions, recall measures the model's ability to detect positive samples, and F1-score is the harmonic mean of precession and recall, reflecting the balance between the two. Support shows the actual number of samples for each class in the dataset. This report is very useful for analyzing model performance, especially in imbalanced datasets or multi-class problems, providing a deeper understanding than just using accuracy.[16]

i. Accuracy

Accuracy is a metric that measures the proportion of correct predictions made by a model, calculated using the formula accuracy = (Number of correct predictions/total data) x 100%. This percentage reflects how well the model performed overall by dividing the number of accurate predictions by the total number of observers. [15]

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Amount of data}} x\ 100\% \quad (1)$$

ii. Precision

Precision in the classification report is an evaluation metric that measures the model's ability to avoid making false positive predictions. precision is calculated as the ratio of the number of correct predictions (True Positive) to the total number of positive predictions (True Positive + False Positive).[16]

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)+False Positives (FP)}} \quad (2)$$

iii. Recall

Recall is an evaluation metric used to measure the proportion of data from a particular class that is successfully classified correctly by the model. Recall is calculated by dividing the number of true positives (TP) by the total number of actual positive data, which is the sum of true positives (TP) and false negatives (FN).[16]

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)+False Negatives (FN)}} \quad (3)$$

iv. F1-score

F1-score is calculated by first multiplying the precision and recall, then dividing the result by the sum of the precision and recall. Finally, the value is multiplied by 2 to obtain the harmonic mean of the precision and recall.[16]

$$\text{F1-Score} = 2 \,.\, \frac{\text{Precision . Recall}}{\text{Precesion + Recall}} \quad (4)$$

4. Training Time

In this paper, the model training time was recorded at 148.92 seconds, with the details of the first epoch taking 81 seconds and the second epoch 67 seconds. The training time in the first epoch was longer due to initial overhead, such as data initialization and caching. Meanwhile, in the second epoch, the training time became shorter due to data pipeline optimization and computational efficiency after the model tensor structure was initialized.

5. Inference Time

In this study, the inference time was recorded at 3.0563 seconds, with an average of 10 ms per step. The inference time is faster than the training time because it only involves the forward pass process without any gradient or backpropagation computation, so the use of computing resources is lighter and more efficient.

6. Device Specifications and Experimental Environment

Experiments were conducted using Google Colab without GPU acceleration, so the entire training and inference process was run on the CPU provided by Colab on the same hardware. In addition, experiments were also run locally on an ASUS VivoBook X421IAY_M413IA laptop with Windows 10 Home Single Language 64-bit (Build 19045), an AMD Ryzen 3 4300U processor with Radeon Graphics (4 cores, ~2.7GHz), and 8GB of RAM. To ensure reproducible results, the libraries used in model development include TensorFlow 2.18.0, NumPy 1.26.4, Pandas 2.2.2, and Scikit-learn 1.6.1. With this configuration, model training is performed efficiently even without GPU support, but CPU usage may affect training time especially for more complex models or larger datasets.

IV. Result And Discussion

A. Model Performance Comparison

Model performance is evaluated based on accuracy, inference time, and training time. Table III summarizes the key metrics for each model.

TABLE III. Model Performance Comparison

Model	Accuracy	Training Time	Inference Time
CNN Purpose	89%	148.9196 Second	3.0563 Second
CNN From Kaggle	85%	1309.5718 Second	5.612 Second
Naïve Bayes From Kaggle	90%	0.1029 Second	4.0366 Second

As shown in Table I. The Naive Bayes model from Kaggle, achieved the highest accuracy (90%) among the three methods. This can be explained by the results of the n-gram analysis using CountVectorizer on the code which shows that the dataset is dominated by unigrams and bigrams with frequently occurring word patterns, such as "chatgpt", "is", "with", and "you". The frequency of occurrence of these words is very high, indicating that the features in the text are quite separate and have a clear probabilistic distribution.

In this study, the author processed 219.294 data, which can be said to be quite large data so that on the one hand the CNN model designed by the author has no reason to achieve accuracy below the naive bayes from Kaggle. However, even though the data processed is quite a lot, the dataset processed is a tweet dataset which as we already know that the tweet dataset is data containing short tweet texts where short texts are easier to classify with naive bayes. If the text in the dataset consists of relatively short sentences, then the patterns that emerge will be easily captured by probabilistic approaches such as naive bayes compared to CNN which requires more complex spatial patterns to extract features in depth.

after that there is a CNN model designed by the author achieving an accuracy of 89% which is slightly better than CNN from Kaggle which is at 85%. The main differences that cause the author's CNN to perform better than the Kaggle CNN lie in the model architecture and training parameters. The author's CNN uses GlobalMaxPooling1D, which is more effective in capturing important features from the entire sequence compared to MaxPooling1D used by the Kaggle CNN. In addition, the author's CNN has an additional Dense layer before the output, which allows the model to learn more complex feature representations before making a final decision. In terms of training, the author's CNN uses a smaller batch size (128) and only 2 epochs, which may help the model avoid overfitting, while the Kaggle CNN uses a batch size of 1024 and 10 epochs, which may cause the model to be less flexible in capturing pattern variations in the data. In addition to accuracy, the training time and inference time of each model are analyzed to evaluate its computational efficiency. As shown in Table I, the CNN model designed by the author shows significant improvements in both training time and inference time compared to the CNN model from Kaggle.

The training time of the author's CNN model is 148.9196 seconds, which is almost 9 times faster than the training time of the CNN model from Kaggle which reaches 1309.5718 seconds. The reasons for the difference in training time between the CNN from Kaggle and the author's CNN are due to several factors. The author's CNN is only trained for 2 epochs, while the Kaggle CNN uses up to 10 epochs, which directly increases the training duration. In addition, the CNN architecture from Kaggle is more complex with additional layers such as MaxPooling1D and Flatten, while the author's CNN uses the lighter GlobalMaxPooling1D.

Similarly, in terms of inference time, the author's CNN model produces the fastest results among the three methods, taking only 3.0563 seconds to make a prediction. In comparison, the CNN model from Kaggle has an inference time of 5.612 seconds and the Naive Bayes model from Kaggle has an inference time of 4.0366 seconds, which are much slower. This increase in inference efficiency further underlines the superiority of the author's CNN model over the three existing models, which are much better.

Overall, the author's CNN model shows an optimal balance between training speed, inference speed, and accuracy, making it the most efficient and practical solution for model implementation.

B. Confusion Matrix Convolutional Neural Network Purpose

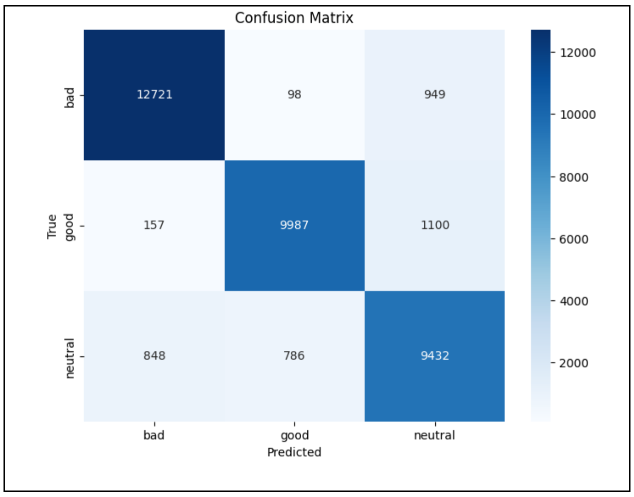
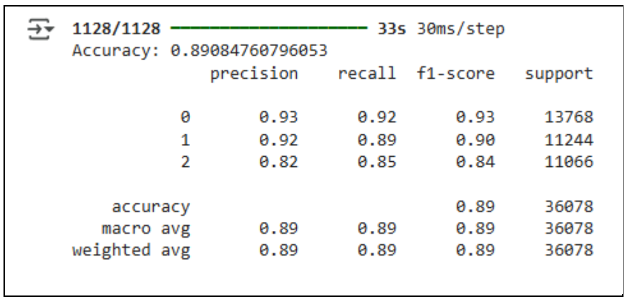


Fig. 5 Confusion Matrix CNN Purpose

The Confusion Matrix in Figure 5 shows the performance of the classification model on three classes: bad, good, and neutral, with the number of correct predictions shown on the diagonal values and misclassifications on the off-diagonal values. The model successfully predicted 12,721 bad data, 9,987 good data, and 9,432 neutral data accurately, possibly because the relevant features in the data were successfully identified in the model. However, some errors occurred, such as 949 bad data predicted as neutral, 1,100 good data predicted as neutral, and 848 neutral data predicted as bad data, which were likely caused by feature similarities between classes, data imbalance, or suboptimal separation of class boundaries in the feature space.

C. Classification Report CNN Purpose



```
1128/1128 ━━━━━━━━━━━━━━━━━━━━ 33s 30ms/step
Accuracy: 0.89084760796053
              precision    recall  f1-score   support

           0       0.93      0.92      0.93     13768
           1       0.92      0.89      0.90     11244
           2       0.82      0.85      0.84     11066

    accuracy                           0.89     36078
   macro avg       0.89      0.89      0.89     36078
weighted avg       0.89      0.89      0.89     36078
```

Fig. 6 Classification Report CNN Purpose

The classification report in Figure 6 shows that the model has an accuracy of 89%, which means that 89% of the total data is successfully predicted correctly. The best performance is achieved in class 0, with precession, recall, and F1-Score of 0.93 each, indicating that the model is very accurate in recognizing and predicting data in this class. Conversely, the lowest performance is in class 2, with precession of 0.82 and recall of 0.85, indicating that there are more errors in predicting data in this class. This difference in performance between classes is likely due to an imbalance in the amount of data or similarity of features between classes, so that the model has difficulty in optimally processing the data. However, the average F1-score value of 0.89 indicates that the model has a good balance between precession and recall overall.

With 89% accuracy this model is very suitable for use in non-critical applications such as sentiment analysis, product recommendations, or early decision support systems. This accuracy provides fairly reliable results for these tasks without major risks. However, if used for more sensitive applications such as medical diagnosis or security systems, performance improvements are needed through model optimization, the addition of more representative data, or balancing techniques so that performance is consistent across all classes. With these improvements, the model can be more effective for applications that require higher accuracy and consistency between classes.

V. Conclusion and Suggestion

A. Conclusion

The results of this study indicate that the author's CNN model has advantages in terms of computational efficiency compared to other models. Although Naive Bayes achieved the highest accuracy (90%) due to its advantage in recognizing word patterns probabilistically, the author's CNN still showed optimal performance with an accuracy of 89%, higher than Kaggle's CNN (85%). In addition, the author's CNN has the fastest inference time (3.0563 seconds), outperforming Kaggle's CNN (5.612 seconds) and Naive Bayes (4.0366 seconds). This advantage is due to the use of GlobalMaxPooling1D, which significantly reduces the feature dimension, as well as large batch data processing (128 data at once), which increases efficiency in running predictions in real time.

In terms of training, the author's CNN is also more efficient than the Kaggle CNN, with a training time of only 148.9196 seconds compared to 1309.5718 seconds on the Kaggle CNN. Meanwhile, Naive Bayes has the fastest training time (0.1029 seconds) because it does not use the backpropagation process, but only calculates the probabilistic distribution of the data features. However, despite being fast in training, Naive Bayes is still inferior in inference speed compared to the author's CNN. Thus, this study confirms that in selecting a sentiment analysis model, the balance between accuracy and efficiency is an important factor.

B. Suggestion

This study can further explore a hybrid approach that combines the efficiency of naive Bayes with the feature extraction capabilities of CNN. to improve the performance of sentiment analysis. In addition, CNN architecture optimization through hyperparameter tuning or the use of pre-trained embedding can help improve training efficiency and accuracy. Future studies can also expand the data coverage by considering language variations and contexts in user tweets to ensure the model's reliability across real-world scenarios.

References

[1] P. P. Wulan and H. Basri, “Analisis Sentimen Terhadap Layanan Nasabah Bank Menggunakan Teknik Klasifikasi Naive Bayes Sentiment Analysis of Banking Customer Service Using Naive Bayes Classification Technique,” *Jurnal Kecerdasan Buatan dan Teknologi Informasi*, vol. 3, no. 2, pp. 68–74, 2024.

[2] F. Fitriyah, U. T. Madura, C. Fitria, and U. T. Madura, “Analisis Sentimen Tentang Ulasan Chatgpt Di Twitter Menggunakan Metode Multinomial Naïve Bayes , Decision Tree , Dan Logistic Regression,” no. December, 2023.

[3] B. Purbayanto and T. N. Suharsono, “Analisis Sentimen Pengguna X terhadap Chatgpt dengan Algoritme Naive Bayes,” *Jurnal Telematika*, vol. 18, no. 2.

[4] L. Yusuf and S. Masripah, “SENTIMEN ANALISIS CHATGPT DENGAN ALGORITMA NAÏVE BAYES DAN OPTIMASI PSO,” *INTI Nusa Mandiri*, vol. 18, no. 1, pp. 59–64, Aug. 2023, doi: 10.33480/inti.v18i1.4230.

[5] S. N. Listyarini and D. A. Anggoro, “Analisis Sentimen Pilkada di Tengah Pandemi Covid-19 Menggunakan Convolution Neural Network (CNN),” *Jurnal Pendidikan dan Teknologi Indonesia*, vol. 1, no. 7, pp. 261–268, 2021, doi: 10.52436/1.jpti.60.

[6] K. Muludi, M. Naufal Humam, D. A. Shofiana, A. Syarif, J. I. Komputer, and U. Lampung, “Perbandingan Kinerja CNN dan Naïve Bayes pada Analisis Sentimen Performa Manchester United di Twitter,” 2023.

[7] D. Transiska, D. Febriawan, and F. N. Hasan, “Analisis Sentimen Terhadap Penggunaan Chatgpt Berdasarka Twitter Menggunakan Algoritma Naïve Bayes,” *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 8, no. 2, p. 1077, Apr. 2024, doi: 10.30865/mib.v8i2.7540.

[8] D. Panca Wilie, “Analisis Sentimen Opini Publik Terhadap Chatgpt di Twitter Menggunakan Metode Naive Bayes,” 2023. [Online]. Available: https://www.kaggle.com/datasets/charunisa/ChatGPT-sentiment-analysis.

[9] E. Y. Hidayat and D. Handayani, “Penerapan 1D-CNN untuk Analisis Sentimen Ulasan Produk Kosmetik Berdasar Female Daily Review,” *Jurnal Nasional Teknologi dan Sistem Informasi*, vol. 8, no. 3, pp. 153–163, 2023, doi: 10.25077/teknosi.v8i3.2022.153-163.

[10] J. M. Polgan *et al.*, “Optimalisasi Algoritma Naive Bayes Dengan Teknik Ensemble Dalam Analisis Sentimen Twitter Pantai Kartini Jepara,” vol. 13, pp. 1331–1341, 2024.

[11] D. F. Rahman, “Analisis Chatgpt Tweet Menggunakan Eda Dan Sentiment Analysis: Data Pengguna Twitter Di Indonesia,” no. December, pp. 60–70, 2023, [Online]. Available: https://www.researchgate.net/publication/376356578

[12] B. Hakim, “Analisa Sentimen Data Text Preprocessing Pada Data Mining Dengan Menggunakan Machine Learning,” *JBASE - Journal of Business and Audit Information Systems*, vol. 4, no. 2, 2021, doi: 10.30813/jbase.v4i2.3000.

[13] M. Metode, “Analisis Sentimen Berbasis Aspek pada EDOM Pembelajaran Aspect-Based Sentiment Analysis in EDOM Learning Using CNN and Word2vec Methods,” vol. 12, no. 3, 2024, doi: 10.26418/justin.v12i3.75610.

[14] T. Leonardo, V. H. Pranatawijaya, P. Bagus, and A. Anugrah, “Analisis Sentimen Mahasiswa terhadap Website PKKMB 2024 Universitas Palangkaraya dengan Metode Machine Learning,” vol. 7, no. 5, pp. 1148–1155, 2024.

[15] K. Kevin, M. Enjeli, and A. Wijaya, “Analisis Sentimen Pengunaaan Aplikasi Kinemaster Menggunakan Metode Naive Bayes,” *Jurnal Ilmiah Computer Science*, vol. 2, no. 2, pp. 89–98, Jan. 2024, doi: 10.58602/jics.v2i2.24.

[16] A. K. Iman, E. Iman, H. Ujianto, F. Sains, and U. T. Yogyakarta, “Analisis Sentimen Pemindahan Ibu Kota Indonesia Menggunakan K-Nearest Neighbor Sentiment Analysis of the Relocation of Indonesia ’ s Capital Using K-Nearest Neighbor,” vol. 4, no. 12, pp. 759–768, 2024.