Terbit online pada laman web jurnal: **http://journal.isas.or.id/index.php/JACOST**

JOURNAL OF APPLIED COMPUTER SCIENCE AND TECHNOLOGY (JACOST)

Vol. 6 No. 1 (2025) 17 – 22 | ISSN: 2723-1453 (Media Online)

Performance Comparison of Support Vector Machine (SVM) and k-Nearest Neighbors (kNN) in Verifying Material Orientation

Eldio Utama[1], Eko Rudiawan Jamzuri[2]
[1,2]Department of Electrical Engineering, Politeknik Negeri Batam, Batam, Indonesia
[1]eldioutama1710@gmail.com, [2]ekorudiawan@polibatam.ac.id *

Abstract

In automated manufacturing, verifying material orientation is essential to ensure the product assembly proceeds without errors. For instance, in the beverage industry, incorrect orientation of materials, such as bottle caps, can lead to failures in the packaging process, resulting in improperly sealed bottles that may compromise product quality and safety. This study compares the performance of Support Vector Machine (SVM) and k-Nearest Neighbors algorithms for verifying material orientation verification through automated optical inspection. The images were processed using the Inception V3 Convolutional Neural Network (CNN) to extract relevant image features, which were then classified using SVM and kNN algorithms. As a result, SVM achieved high classification performance during testing, with classification accuracy, precision, recall, and F1 score of 1.0 compared to kNN, which achieved only 0.967. However, kNN demonstrated superior computational efficiency, with a training time of 1.126 seconds and a validation time of 0.713 seconds, compared to SVM's training time of 3.101 seconds and validation time of 1.479 seconds. These results indicate that while both methods are highly effective for material orientation verification, kNN offers significant advantages in terms of computational speed, making it more suitable for real-time applications. The implications of this study highlight the potential for integrating the proposed method in industrial applications, promoting enhanced efficiency and reducing error rates in automated assembly lines.

Keywords: Automated Optical Inspection, Inception V3, Convolutional Neural Network, Support Vector Machine, k-Nearest Neighbors

1. Introduction

Material orientation plays a crucial role in ensuring the success of product installation and packaging processes. Incorrect orientation of materials can disrupt these processes, leading to failures. For instance, in the automated packaging of beverage products, bottle caps must be correctly positioned and oriented to ensure a seamless operation. Typically, this orientation process is facilitated by a bowl feeder, a vibrating system designed to orient and feed small parts automatically within the manufacturing line [1]. Although effective, the bowl feeder system is not infallible; occasionally, bottle caps remain incorrectly oriented on the conveyor, resulting in failures during the packaging stage. Such orientation errors not only disrupt the packaging process but can also compromise product quality and increase operational costs. To illustrate this issue, a visual depiction of an incorrectly oriented bottle cap after the bowl feeder process is provided in Figure 1, highlighting the potential for orientation errors that can lead to misalignment and failure in subsequent stages.

A sorting process using sensors is commonly implemented to detect and address material orientation errors, ensuring only correctly oriented materials proceed to the next production stage. This sorting mechanism aims to remove improperly oriented items from the line, thus preventing disruptions. In specific applications, proximity sensors are used. For instance, prior research in [2] employed photoelectric proximity and color sensors to sort red and green apples, while other studies have utilized inductive proximity sensors to separate specific trash [3] and plastic and metal beverage bottles in another [4].

While proximity sensors can effectively sort defective materials, they are not suitable for specific items such as bottle caps. The distinct shape of bottle caps poses a challenge for detecting orientation errors using conventional proximity sensors. Accurate orientation checks can only be achieved through visual inspection, typically by observing the top surface of the bottle cap. This limitation motivates the current research, which aims to verify bottle cap orientation using visual techniques. In this approach, images of bottle caps are captured from above using a camera sensor. A pattern recognition process is then applied, leveraging machine learning algorithms to predict whether the bottle cap orientation is correct. Pattern recognition involves classifying input data into items and categories based on defined characteristics, as described in [5].

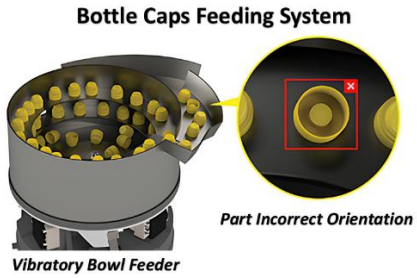


Figure 1. Bottle caps feeding system and possible defect

Support Vector Machines (SVM) and k-Nearest Neighbors (kNN) are commonly studied to assess their effectiveness in various specific applications. For instance, prior research in [6] compared and analyzed classification models for image forgery detection, although it involved complex feature extraction techniques. Research in [7] also highlights the performance of SVM and kNN when classifying handwritten digits. However, the performance assessment is based solely on classification accuracy. It does not emphasize the importance of other classification metrics, such as precision, recall, and F1 score, for a more comprehensive evaluation.

Beyond verifying material orientation, this research aims to compare the performance of SVM and kNN algorithms in this specific application. Additionally, a feature extraction method using Inception V3 image embedding is proposed to enhance the accuracy and robustness of verifying material orientation. Performance comparison in this study is conducted using evaluation metrics such as confusion matrix, precision, accuracy, F1 score, and recall, ensuring a comprehensive assessment. Moreover, comparison in terms of computational efficiency during training and validation is evaluated. The findings are intended to contribute toward the development of a material orientation verification system for production lines.

This paper is organized as follows: Section 2 discusses the research methods, including the image embedding technique using Inception V3 and the application of SVM and kNN algorithms for classification. Section 3 presents the evaluation results, offering a comprehensive analysis of each model's performance. Finally, Section 4 provides conclusions and recommendations for future research, highlighting potential avenues for improving accuracy and efficiency in material orientation verification systems.

2. Materials and Methods

This research was conducted through three main stages. The first stage involved data collection of bottle caps. The second stage was a feature extraction, in which features were extracted from each image using the Inception V3 image embedding. The third stage focused on classification and evaluation, comparing the performance of the SVM and kNN.

2.1. Dataset Collection

The dataset used in this study consists of 1950 images of bottle caps, with 975 labeled as having the correct orientation and 975 labeled as having the incorrect orientation. The dataset is split into two subsets: 1500 images for training and validation and the remaining 450 images for testing. Data on bottle caps with five color variations—white, blue, red, orange, and green—were collected to evaluate whether the proposed method can be applied to bottle caps of different colors. Images were captured using a Basler industrial camera and were taken under three lighting conditions: bright, dim, and low light to enhance data variety. Sample images illustrating correct and incorrect orientations under these conditions are shown in Figure 2. Specifically, Figure 2 (a), (b), and (c) display bottle caps under bright lighting, dim lighting, and low-light indoor settings, where minimal light from adjacent rooms creates dim conditions.

2.2. Image Embedding

Image embedding is a feature extraction method used in this study to create high-dimensional vector representations of images. The purpose of image embedding is to organize visual information efficiently, allowing the model to capture essential features from images [8]. The Inception V3 Convolutional Neural Network (CNN) architecture is used as the basis for image embedding to extract complex image features. Through the use of Inception V3, detailed visual patterns essential for material orientation verification are captured.

A pipeline was designed using Orange Data Mining to streamline the workflow. Orange Data Mining is an open-source data visualization and analysis tool that provides a visual programming environment. It allows users to connect modular widgets for different processing stages, making it easier to manage each step in the classification process. Figure 3 illustrates the workflow for verifying the bottle cap orientation.

The first step involves loading images from the dataset folder. The dataset folder contains two subfolders: Correct Orientation and Incorrect Orientation. Each subfolder contains categorized images based on orientation. The second step is performing feature extraction, which extracts feature vector representations of each image. Here, Inception V3 image embedding is used.

Inception V3 is a CNN architecture that combines multiple convolutional filters to enhance feature extraction across different scales. Inception V3 employs regularization and dropout techniques to avoid overfitting. Compared to Inception V2, this architecture offers smaller grid sizes and improvement of convolution efficiency. The improvement in Inception V3 can reduce computational resources while maintaining high accuracy [9].

DOI : https://doi.org/10.52158/jacost.v6i1.1037

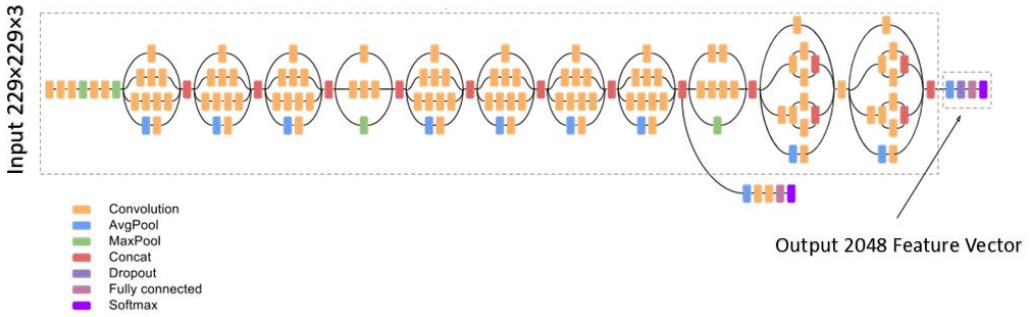


Figure 2. Inception v3 image embedding architecture.

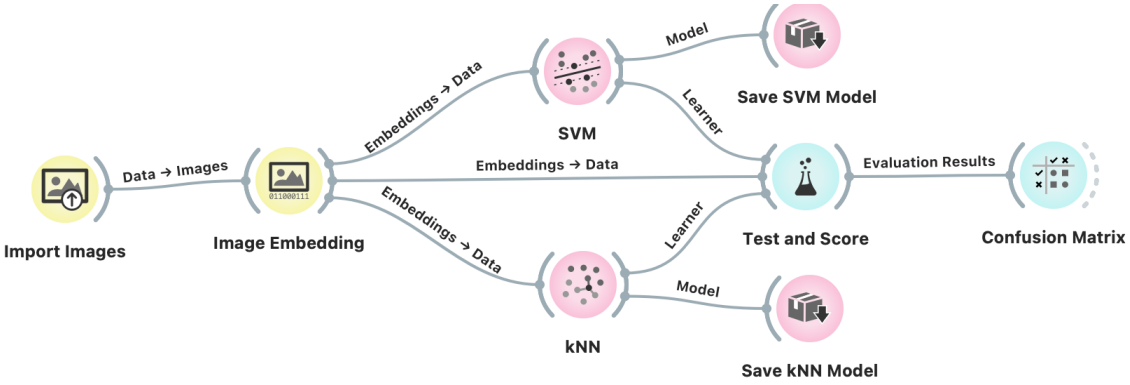


Figure 4. Classification workflow on Orange3 data mining software.

In the feature extraction process, each input image is resized to 299×299 pixels to match the Inception V3 input dimensions. After resizing, the images are processed through the entire Inception V3 architecture, as shown in Figure 4. The architecture consists of multiple stages, including an input layer, where initial convolutional and max-pooling handle basic preprocessing. Next, the images pass through inception modules A, B, and C. Each is repeated multiple times with different configurations to capture features at varying complexity levels. Between these modules, grid size reduction layers decrease the spatial dimensions of the feature maps, facilitating deeper feature extraction in subsequent layers. After that, the images pass through the last inception module to capture high-level, abstract features. The feature extraction process yields the output of a 2048-dimensional vector for each image. These vectors are compressed yet rich representations containing patterns, textures, shapes, and other high-level features that the Inception V3 neural network has extracted.

2.3. Support Vector Machine

Support Vector Machine (SVM) is a supervised learning algorithm that classifies data using a linear function in high-dimensional feature space [10]. SVM is trained on labeled data to find an optimal separating hyperplane, a boundary in feature space that maximally divides classes. The purpose of the SVM training is to maximize the margin or distance between the classes on either side of the hyperplane. In cases where data cannot be linearly separated, SVM can leverage various kernel functions, such as linear, polynomial, and Radial Basis Function (RBF) kernels. These kernel functions are used to map data into higher-dimensional spaces and apply non-linear classification.

SVM has several tuning parameters, including Cost (C) and Epsilon (ϵ). The regularization parameter C controls the trade-off between maximizing the margin and minimizing classification errors. A higher value of C allows the model to avoid misclassification and regression errors [11]. The parameter ε specifies a threshold within which errors are ignored when fitting the hyperplane. In this study, the parameters were set to C=1.0 and ϵ=0.10 to allow slight deviations without penalizing the model significantly.

A linear kernel was selected to compute the dot product between two feature vectors, serving as similarity measurement. Equation 1 describes the formula for calculating linear kernel. Function $K(x, y)$ represents the kernel value between vectors x and y. Notation x_i and y_i are the feature values at the i-th index, and n is the input vector dimension.

$$K(x, y) = \sum_{i=1}^{n} x_i \cdot y_i \quad (1)$$

SVM computes the linear kernel and then uses the resulting kernel values to find the optimal hyperplane. A

DOI : https://doi.org/10.52158/jacost.v6i1.1037

higher dot product suggests closer similarity in feature space. In contrast, a lower value indicates more significant dissimilarity. Consequently, this approach enables SVM to classify images based on their embedded feature vectors effectively, allowing for robust separation between classes.

2.3. k-Nearest Neighbors

k-Nearest Neighbors (kNN) is a classification algorithm that assigns a data point to a class based on the most common class among its closest neighbors. In this method, k is a user-defined parameter representing the number of nearest neighbors considered during the classification process. kNN works by identifying the closest data points to the new data point being classified and then assigning its class based on the majority class of those neighbors [12]. The parameter k was set k=5, meaning that each data point will be classified based on the five nearest neighbors. This classification relies on the similarity between the new data and existing data in the feature space.

The similarity is measured by Euclidean distance, which calculates the distance between two points in cartesian space. Equation 2 shows the formula for calculating Euclidean distance. Notation $d(p, q)$ represents the distance between points p and q, with p_i and q_i as the coordinates of those points, and notation n defines the number of dimensions of the input vector. Once these distances are computed, the k nearest neighbors are identified, and the classification process proceeds.

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + \cdots + (p_n - q_n)^2} \quad (2)$$

Moreover, a uniform weighting scheme was applied for kNN that assigns equal weight to all k nearest neighbors, regardless of their specific distance from the target data point. Consequently, each neighbor contributes equally to determining the class of the new data point. By using a uniform weighting method and Euclidean distance metric, this kNN model provides a straightforward yet effective approach to classifying high-dimensional feature vectors derived from image embedding, thus enhancing the accuracy of material orientation verification.

2.3. Performance Evaluation

Algorithm evaluation begins with K-Fold cross-validation, a method that estimates prediction errors and assesses model performance. In K-Fold cross-validation, the dataset is divided into k subsets of approximately equal size. The classification model is trained and validated k times, with each iteration using one subset as validation data while the remaining subsets serve as training data. Cross-validation, also known as rotation estimation, helps determine the robustness of the model across different data segments [13]. In this study, 10-fold cross-validation is applied to 1500 samples. In each iteration, the data is split into 1350 training samples and 150 validation samples.

Table 1. Comparison results of SVM and kNN on verifying material orientation during training and validation.

Model	CA	Prec.	Recall	F1 score	Training Time	Validation Time
SVM	1.0	1.0	1.0	1.0	3.101	1.479
kNN	1.0	1.0	1.0	1.0	1.126	0.713

Table 2. Comparison results of SVM and kNN on verifying material orientation during testing.

Model	CA	Prec.	Recall	F1 score
SVM	1.0	1.0	1.0	1.0
kNN	0.967	0.969	0.967	0.967

The performance of SVM and kNN is compared using several evaluation metrics, including confusion matrix, classification accuracy (CA), precision, recall, and F1 score. The confusion matrix presents the classification results with four key variables: True positive (TP), true negative (TN), false positive (FP), and false negative (FN). TP represents cases where both the actual and predicted classes are positive, and TN indicates cases where both are negative. FP occurs when a negative class is incorrectly predicted as positive, while FN arises when a positive class is incorrectly predicted as negative. CA is the ratio of correct predictions to the total number of samples, calculated as Equation 3. Precision is the proportion of true positive predictions out of the total positive predictions, calculated by Equation 4. Recall is the ratio of true positive predictions to the total number of actual positive examples, calculated as Equation 5. The F1 score balances precision and recall, providing a harmonic mean of the two, calculated by Equation 6.

$$\text{CA} = \frac{TP + TN}{TP + FP + FN + TN} \quad (3)$$

$$\text{Precision} = \frac{TP}{TP + FP} \quad (4)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (5)$$

$$\text{F1 score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (6)$$

By analyzing these metrics, a comprehensive understanding of each model's performance in handling part orientation verification is obtained. A higher accuracy, precision, recall, and F1 score indicate better classification performance, contributing to a reliable assessment of the methods used in this study.

3. Results and Discussion

This section outlines the performance evaluation of the classification model implemented on the bottle caps dataset. The analysis encompasses the model's capability to validate bottle cap orientation, utilizing the confusion matrix derived from the SVM and kNN methodologies.

DOI : https://doi.org/10.52158/jacost.v6i1.1037

Furthermore, the evaluations are detailed to obtain classification accuracy, precision, recall, and F1 score.

3.1. Classification and Computational Efficiency

Table 1 shows the results of a validation process comparing the performance of SVM and kNN methods. The evaluation was based on classification metrics, including classification accuracy (CA), precision, recall, F1 score, and training and validation time.

Both SVM and kNN achieved perfect scores of 1.0 across all classification metrics: CA, precision, recall, and F1 score. This result indicates that both methods were highly effective in classifying the dataset, demonstrating their ability to achieve flawless predictions without any misclassifications.

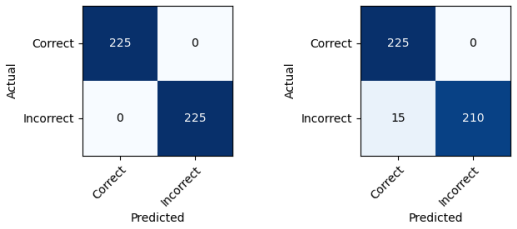
However, differences were observed in computational efficiency, specifically in the training and validation time. For SVM, the training process took 3.101 seconds, while the validation phase required 1.479 seconds. In contrast, kNN demonstrated significantly faster performance, with a training time of 1.126 seconds and a validation time of 0.713 seconds. This result highlights kNN's computational efficiency, especially in scenarios where rapid validation and training are critical.

Table 2 compares the performance of SVM and kNN in verifying bottle cap orientation during the testing phase. SVM once again achieved perfect performance, registering 1.0 in accuracy, precision, recall, and F1 score. Meanwhile, kNN performed reliably, recording 0.967 in accuracy, 0.969 in precision, 0.967 in recall, and 0.967 in F1 score. Although both methods effectively distinguish correct from incorrect orientations, SVM demonstrated a slight edge by correctly classifying every sample in the test set. This result underscores SVM's robustness and reliability for orientation verification tasks.

In summary, while SVM outperforms kNN in terms of accuracy, precision, recall, and F1 score, kNN offers superior computational speed. These findings suggest that kNN may be more appropriate for real-time applications or scenarios involving large-scale data where processing speed is critical.

3.2. Confusion Matrix

Figure 5 illustrates the confusion matrices for both the SVM and kNN models during testing. In Figure 5 (a), the confusion matrix for SVM shows no misclassifications, indicating that all correctly oriented and incorrectly oriented samples were accurately classified. By contrast, Figure 5 (b) presents the confusion matrix for kNN, revealing that 15 incorrectly oriented samples were erroneously predicted as correctly oriented. Overall, SVM demonstrates superior performance based on the confusion matrix results obtained in the testing phase.



(a) (b)

Figure 5. Resulted confusion matrix from (a) SVM and (b) kNN during testing.

4. Conclusion

In conclusion, this study demonstrates that both SVM and kNN are highly effective for verifying material orientation achieving strong classification metrics. SVM showed marginally better performance during testing, achieving a perfect classification score of 1.0, while kNN achieved 0.967. However, notable differences were observed in computational efficiency. kNN outperformed SVM in terms of computational speed, with significantly faster training and validation times. The training time for kNN was 1.126 seconds compared to 3.101 seconds for SVM, while the validation time for kNN was 0.713 seconds compared to 1.479 seconds for SVM. This result makes kNN more efficient and better suited for real-time applications or scenarios involving larger datasets, where processing speed is critical.

In future research, it is suggested that alternative approaches for evaluating material orientation be explored, such as convolutional neural networks (CNN), gradient boosting, and other machine learning techniques. Expanding the evaluation to include these methods could enable a more comprehensive comparison, particularly with a more extensive and more varied dataset. This approach may offer new insights and allow for a deeper analysis of algorithm performance in material orientation evaluation.

Acknowledgment

We want to thank the Politeknik Negeri Batam for facilitating this research.

References

[1] S. Azhar and S. I. A. Shah, "Modeling and Analysis of a Vibratory Bowl Feeder," in *2021 Seventh International Conference on Aerospace Science and Engineering (ICASE)*, Dec. 2021, pp. 1–13. doi: 10.1109/ICASE54940.2021.9904038.

[2] S. Haque *et al.*, "Automatic Product Sorting and Packaging System," in *2023 15th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, Aug. 2023, pp. 163–167. doi: 10.1109/IHMSC58761.2023.00046.

[3] R. Wulandari, M. R. Ariwibowo, T. Taryo, and G. Ananda, "Design Smart Trash Based on the Inductive Proximity Sensor," *International Journal of Multidisciplinary Approach Research and Science*, vol. 2, no. 01, Art. no. 01, 2024, doi: 10.59653/ijmars.v2i01.394.

[4] A. D. Sevtian, F. A. Kurniawan, Yulfitra, and M. Arifin, "Pemograman Sistem pada Mesin Filling Bottle PLC dengan

Menggunakan Penggerak Pneumatik dan Intelegensi Sensor," *Jurnal MESIL (Mesin Elektro Sipil)*, vol. 3, no. 2, Art. no. 2, Dec. 2022, doi: 10.53695/jm.v3i2.807.

[5] C. Singh, "Machine Learning in Pattern Recognition," *European Journal of Engineering and Technology Research*, vol. 8, no. 2, Art. no. 2, Apr. 2023, doi: 10.24018/ejeng.2023.8.2.3025.

[6] L. Almawas, A. Alotaibi, and H. Kurdi, "Comparative Performance Study of Classification Models for Image-splicing Detection," *Procedia Computer Science*, vol. 175, pp. 278–285, Jan. 2020, doi: 10.1016/j.procs.2020.07.041.

[7] A. Rajput and A. K. Singh, "Handwritten Digit Recognition Accuracy Comparison Using KNN, CNN and SVM," *Educational Administration: Theory and Practice*, vol. 30, no. 2, Art. no. 2, Apr. 2024, doi: 10.53555/kuey.v30i2.1676.

[8] K. B. Narayanan, D. K. Sai, K. A. Chowdary, and S. R. K, "Applied Deep Learning Approaches on Canker Effected Leaves to Enhance the Detection of the Disease Using Image Embedding and Machine Learning Techniques," *EAI Endorsed Transactions on Internet of Things*, vol. 10, Mar. 2024, doi: 10.4108/eetiot.5346.

[9] U. Ungkawa and G. A. Hakim, "Klasifikasi Warna pada Kematangan Buah Kopi Kuning menggunakan Metode CNN Inception V3," *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*, vol. 11, no. 3, Art. no. 3, Jul. 2023, doi: 10.26760/elkomika.v11i3.731.

[10] S. Ulum, R. F. Alifa, P. Rizkika, and C. Rozikin, "Perbandingan Performa Algoritma KNN dan SVM dalam Klasifikasi Kelayakan Air Minum," *Generation Journal*, vol. 7, no. 2, Art. no. 2, Jul. 2023, doi: 10.29407/gj.v7i2.20270.

[11] R. H. Dananjaya, S. Sutrisno, and F. A. Wellianto, "Akurasi Penggunaan Metode Support Vector Machine dalam Prediksi Penurunan Pondasi Tiang," *Matriks Teknik Sipil*, vol. 10, no. 3, Art. no. 3, Dec. 2022, doi: 10.20961/mateksi.v10i3.64519.

[12] A. Singh, M. Singh, and K. Kumar, "A Hybrid Method for Intrusion Detection Using SVM and k-NN," in *Conference Proceedings of ICDLAIR2019*, M. Tripathi and S. Upadhyaya, Eds., Cham: Springer International Publishing, 2021, pp. 119–126. doi: 10.1007/978-3-030-67187-7_13.

[13] D. Alita, Y. Fernando, and H. Sulistiani, "Implementasi Algoritma Multiclass SVM Pada Opini Publik Berbahasa Indonesia di Twitter," *Jurnal Tekno Kompak*, vol. 14, no. 2, Art. no. 2, Aug. 2020, doi: 10.33365/jtk.v14i2.792.

DOI : https://doi.org/10.52158/jacost.v6i1.1037