Optimalisasi Ekstraksi Fitur dan Klasifikasi untuk Deteksi Objek di *IoT*

Vincentius Kevin Nicklaus Rudolf Huizen
Universitas Atma Jaya
e-mail: [1]210711083@students.uajy.ac.id

Diajukan: 6 Oktober 2023; Direvisi: 6 November 2023; Diterima: 8 November 2023

Abstrak

Penelitian ini bertujuan untuk meningkatkan efektivitas deteksi objek pada sistem Internet of Things (IoT) melalui optimalisasi metode ekstraksi fitur dan klasifikasi. Mengetahui kompleksitas algoritma pada sistem deteksi objek merupakan strategi untuk optimasi pada sistem IOT. Metode ekstraksi fitur yang digunakan adalah Gray Level Co-occurrence Matrix (GLCM), dengan fitur tekstur seperti kontras, homogenitas, energi, dan entropi dari suatu objek gambar. Untuk metode klasifikasi yang dikombinasikan dengan GLCM meliputi K-Nearest Neighbors (K-NN), Support Vector Machines (SVM), Decision Trees, dan Neural Networks. Dari hasil pengujian waktu eksekusi berbagai algoritma klasifikasi seperti K-NN, SVM, Decision Trees, dan Neural Networks, terlihat perbedaan yang signifikan dalam efisiensi dan skalabilitas masing-masing algoritma. SVM menunjukkan waktu eksekusi tertinggi dengan pertumbuhan eksponensial (O(n^3)), sehingga kurang efisien dan kurang sesuai untuk dataset yang sangat besar. Untuk K-NN memiliki kompleksitas waktu eksekusi yang linear dalam faktor k (O(nk)), namun masih terdapat peningkatan secara signifikan dengan bertambahnya jumlah data. Decision Trees, dengan kompleksitas log-linear (O(n log n)), menunjukkan keseimbangan yang baik antara efisiensi dan skalabilitas, sehingga model ini sesuai untuk dataset yang lebih besar dibandingkan SVM dan K-NN. Neural Networks menunjukkan sebagai algoritma yang efisien dengan pertumbuhan waktu eksekusi yang paling lambat (O(n)), sehingga model tersebut sesuai untuk dataset besar.

Kata kunci: *Optimalisasi, Ekstraksi fitur, Klasifikasi, Internet of things, Gray Level Co-occurrence Matrix.*

Abstract

This research aims to enhance the effectiveness of object detection in Internet of Things (IoT) systems by optimizing feature extraction and classification methods. Understanding the complexity of algorithms in object detection systems is a strategy for optimizing IoT systems. The feature extraction method used is the Gray Level Co-occurrence Matrix (GLCM), with texture features such as contrast, homogeneity, energy, and entropy of an image object. The classification methods combined with GLCM include K-Nearest Neighbors (K-NN), Support Vector Machines (SVM), Decision Trees, and Neural Networks. The results of execution time testing for various classification algorithms such as K-NN, SVM, Decision Trees, and Neural Networks show significant differences in the efficiency and scalability of each algorithm. SVM exhibits the highest execution time with exponential growth (O(n^3)), making it less efficient and unsuitable for very large datasets. K-NN has a linear time complexity in the factor of k (O(nk)), but still shows a significant increase in execution time as the amount of data increases. Decision Trees, with log-linear complexity (O(n log n)), demonstrate a good balance between efficiency and scalability, making them suitable for larger datasets compared to SVM and K-NN. Neural Networks are shown to be efficient algorithms with the slowest growth in execution time (O(n)), making them suitable for large datasets.

Keywords*: Optimization, Feature extraction, Classification, Internet of Things, Gray Level Co-occurrence Matrix.*

1. Pendahuluan

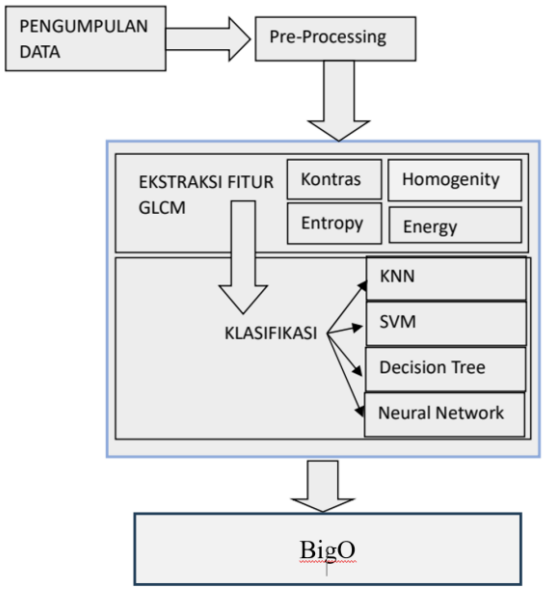
Perkembangan teknologi *Internet of Things* (*IoT*) telah mengubah berbagai aspek industri dan kehidupan sehari-hari. Perangkat *IoT*, seperti sensor dan kamera, memainkan peran penting dalam mengumpulkan data dari lingkungan[1], [2]. Salah satu tantangan utama dalam mempertahankan efektivitas peralatan *IoT* adalah optimalisasi untuk untuk deteksi dengan menggunakan penggabungan ekstraksi fitur

dan klasifikasi. Variabilitas dan kompleksitas data gambar sering kali membuat proses deteksi objek menjadi sulit dan kurang akurat dikarenakan keterbatasan dalam kapasitas *edge computing*[3].

Untuk mengatasi tantangan ini, pemrosesan menggunakan *edge computing* diperlukan metode ekstraksi dan klasifikasi yang efektif. *Edge computing* memungkinkan pemrosesan data dilakukan di dekat sumber data atau perangkat *IoT* itu sendiri[4], [5]. Hal ini dapat mengurangi latensi memungkinkan respons yang lebih cepat serta efisien dalam analisis data. Pendekatan untuk mengatasi tantangan ini adalah dengan mengoptimalkan teknik ekstraksi fitur dan klasifikasi. Gray Level Co-occurrence Matrix (GLCM) merupakan salah satu teknik analisis tekstur yang dapat digunakan untuk mengukur frekuensi pasangan piksel dengan tingkat keabuan tertentu yang terpisah oleh jarak dan sudut tertentu dalam citra[6]. Penyesuaian parameter GLCM, seperti kontras, homogenitas, energi, dan entropi, memungkinkan ekstraksi fitur yang lebih mendalam dan akurat, sehingga meningkatkan kemampuan peralatan *IoT* dalam menginterpretasikan karakteristik objek[7]. Penggunaan metode ekstraksi fitur GLCM yang efektif membutuhkan langkah-langkah yang terstruktur untuk mengukur berbagai fitur tekstur dari gambar yang diperoleh. Kontras mengukur perbedaan intensitas antara piksel yang berdekatan, homogenitas mengevaluasi keseragaman distribusi intensitas, energi mencerminkan keberadaan pola-pola tertentu, dan entropi mengindikasikan tingkat kekacauan dalam citra[8]. Kombinasi fitur-fitur memberikan representasi yang mendalam tentang tekstur objek, yang sangat penting untuk proses klasifikasi[9]. Komposisi ekstraksi dan klasifikasi yang efektif dapat pula mengoptimalkan hasil deteksi. Algoritma klasifikasi yang dipertimbangkan meliputi K-Nearest Neighbors (K-NN), Support Vector Machines (SVM), Decision Trees, dan Neural Networks. K-NN menawarkan metode yang sederhana namun efektif berdasarkan kedekatan dalam ruang fitur, sementara SVM menawarkan keunggulan dalam menemukan hyperplane optimal untuk memisahkan kelas-kelas dalam ruang fitur dengan margin maksimum[10]. Decision Trees menyediakan struktur pohon keputusan yang intuitif dan mudah diinterpretasikan untuk klasifikasi[11], sedangkan Neural Networks menggunakan deep learning yang mampu menangkap pola kompleks dalam data melalui arsitektur berlapis yang memungkinkan pemrosesan fitur secara hierarkis. Pemilihan algoritma klasifikasi yang tepat sangat bergantung pada karakteristik data. Evaluasi menyeluruh terhadap optimasi menggunakan uji kompleksitas Big O. Eksplorasi penelitian ini bertujuan mengembangkan model ekstraksi fitur dan klasifikasi yang efektif untuk meningkatkan deteksi objek dalam sistem *IoT* yang difungsikan sebagai edge computing, sehingga mengetahui kompleksitas algoritma berperan penting untuk performa sistem [12], [13].

2. Metode Penelitian

Metodologi penelitian optimasi metode ekstraksi dan klasifikasi berfokus pada ekstraksi fitur dengan GLCM (kontras, energi, entropi dan homogeniti) dan klasifikasi dengan K-Nearest Neighbors (KNN), Support Vector Machines (SVM), Decision Trees, dan Neural Networks, model tersebut ditunjukkan pada Gambar1.



Gambar 1. Model optimasi ekstraksi fitur dan klasifikasi.

Pada Gambar 1 menunjukkan tahapan alur penelitian dimulai dengan pengumpulan data, pre-processing, yang terdiri atas proses resizing gambar, normalisasi, dan penghilangan noise pada data gambar. Tahap berikutnya fitur-fitur tekstur dari gambar diekstraksi menggunakan metode Gray Level Co-occurrence Matrix (GLCM), dengan fitur kontras, homogeniti, entropi, dan energi. Setelah fitur-fitur ini diekstraksi tahapan berikutnya proses klasifikasi. Pada tahap ini klasifikasi menggunakan algoritma K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree, dan Neural Network. Hasil dari proses klasifikasi ini kemudian dievaluasi dengan menggunakan kompleksitas algoritma (Big O). Kompleksitas algoritma untuk masing-masing metode ditunjukkan pada Tabel 1.

Tabel 1. Kompleksitas algoritma metode ekstraksi dan klasifikasi.

Metode	Rumus	Big O
Ekstraksi Fitur GLCM	$Contrast = \sum_i \sum_j (i-j)^2 * P(i,j)$ $Homogeneity = \sum_i \sum_j \frac{P(i,j)}{1+(i-j)^2}$ $Energy = \sum_i \sum_j P(i,j)^2$ $Entropy = -\sum_i \sum_j P(i,j) * log(P(i,j))$	$O(n^2)$
K-Nearest Neighbors (K-NN)	$d(p,q) = \sqrt{\sum_{i=1}^{d} (q_i - p_i)^2}$	$O(n \times k)$
Support Vector Machines (SVM)	$f(x) = \text{sgn}\left(\sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b\right)$	$O(n^3)$
Decision Trees	$\text{Entropy}(S) = -\sum_{i=1}^{c} p_i \log_2 p_i$	O(nlogn)
Neural Networks	$y = \sigma\left(\sum_{j=1}^{m} w_j \sigma\left(\sum_{i=1}^{d} w_{ij} x_i + b_j\right) + b\right)$	$O(n)$

Pada tabel 1 menunjukkan metode Gray Level Co-occurrence Matrix (GLCM) yang digunakan untuk analisis tekstur dalam citra, dengan mengevaluasi hubungan piksel yang berdekatan. Metode ini mengukur aspek-aspek seperti kontras, yang mengevaluasi perbedaan tingkat keabuan antar piksel. Untuk homogenitas, mengukur kedekatan tingkat keabuan piksel-piksel. Menghitung energi pada piksel dengan mengevaluasi konsistensi tekstur dalam citra. Untuk entropi menilai berdasarkan keragaman atau acak piksel. Berdasarkan proses evaluasi dari metode GLCM maka kompleksitas komputasional dari GLCM sebesar $O(n^2)$, menunjukkan bahwa waktu yang dibutuhkan untuk proses ekstraksi akan meningkat secara kuadrat dengan peningkatan ukuran data citra [14], [15]. Pada metode klasifikasi K-Nearest Neighbors (K-NN) mengidentifikasi kategori sebuah titik data berdasarkan kategori yang paling sering muncul di antara (k) tetangga terdekatnya. Algoritma ini menghitung jarak antar titik dalam ruang fitur dan mengklasifikasikan titik data berdasarkan kedekatan jarak tersebut. Kompleksitas waktu dari K-NN adalah O(n x k), yang bergantung pada jumlah data (n) dan jumlah tetangga (k). Ini berarti bahwa kompleksitasnya meningkat secara linier dengan ukuran dataset dan jumlah tetangga yang dipertimbangkan. Sedangkan untuk Support Vector Machines (SVM) merupakan metode yang dirancang untuk menemukan hyperplane terbaik yang memisahkan dataset menjadi kelas-kelas dengan margin terlebar yang mungkin. SVM menggunakan konsep vektor pendukung, yang adalah titik-titik data yang paling dekat dengan hyperplane, titik-titik ini membantu menentukan orientasi dan posisi hyperplane tersebut. Kompleksitas waktu dari SVM umumnya $O(n^3)$. Decision Trees mengklasifikasikan data dengan membuat struktur pohon, di mana setiap node mewakili sebuah atribut dan setiap cabang mewakili keputusan, yang secara efektif memisahkan data berdasarkan atribut-atribut yang memiliki informasi terbanyak. Pohon keputusan ini

berguna dalam klasifikasi karena mudah dipahami dan diimplementasikan. Kompleksitas waktu Decision Trees yaitu (O(nlog n), menunjukkan bahwa waktu yang diperlukan untuk membangun pohon berkembang logaritmik terhadap jumlah tahapan yang diurutkan. Neural Networks menggunakan lapisan dari neuron yang terhubung untuk meniru proses pengambilan keputusan dalam otak manusia [14], [16]. Jaringan ini belajar dari data latih, di mana bobot dan bias dari setiap neuron disesuaikan untuk meminimalkan kesalahan dalam prediksi. Kompleksitas waktu dari neural networks biasanya adalah O(n), di mana (n) adalah jumlah total operasi per lapisan. Namun, faktor lain seperti jumlah lapisan dan neuron per lapisan juga dapat berpengaruh signifikan pada waktu komputasi. Alur penelitian ditunjukkan pula dalam pseudocode Tabel 2.

Tabel 2. Pseudocode Model Optimasi dengan ekstraksi dan klasifikasi.

Input	data = kumpulkanData()
Output	BigO = hitungBigO(result)

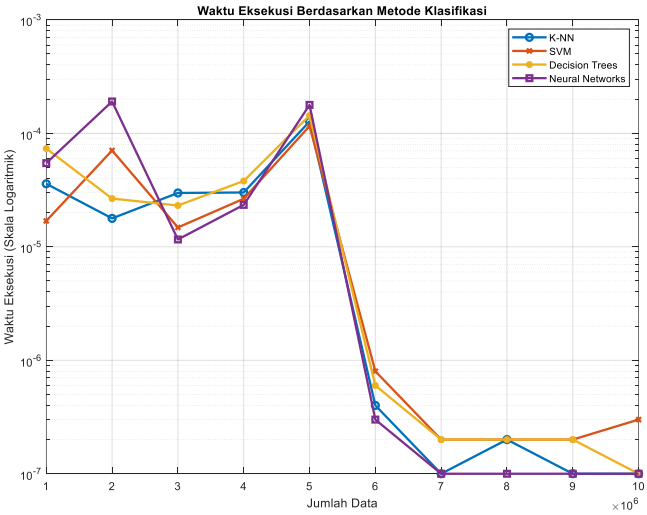
Proses	preprocessedData = preProcessing(data)
	features = ekstraksiFiturGLCM(preprocessedData)
	kontras = 0
	homogeneity = 0
	energy = 0
	entropy = 0
	for i from 1 to n:
	  for j from 1 to n:
	  kontras += (i - j)^2 * P(i,j)
	  homogeneity += P(i,j) / (1 + (i - j)^2)
	  energy += P(i,j)^2
	  if P(i,j) != 0:
	  entropy -= P(i,j) * log(P(i,j))
	resultKNN = []
	for each point in features:
	  d(p,q) = sqrt(sum((qi - pi)^2 for i from 1 to d))
	  resultKNN.append(d(p,q))
	resultSVM = []
	for each point in features:
	  f(x) = sgn(sum(alpha_i * y_i * K(x_i, x) + b for i from 1 to n))
	  resultSVM.append(f(x))
	resultDecisionTree = []
	for each point in features:
	  S = -sum(p_i * log2(p_i) for i from 1 to c)
	  resultDecisionTree.append(S)
	resultNeuralNetwork = []
	for each point in features:
	  y = sigma(sum(w_j * sigma(sum(w_ij * x_i + b_j) + b) for j from 1 to m))
	  resultNeuralNetwork.append(y)

3. Hasil dan Pembahasan

Berdasarkan hasil pengujian ditunjukkan pada Gambar 2, waktu eksekusi dari berbagai algoritma klasifikasi (K-NN, SVM, Decision Trees, dan Neural Networks) ditunjukkan bahwa kompleksitas waktu komputasi algoritma SVM menunjukkan paling tinggi dengan pertumbuhan eksponensial (O(n^3)). Waktu komputasii SVM meningkat sangat drastis seiring dengan pertambahan jumlah data. Sedangkan untuk K-NN memiliki kompleksitas waktu komputasi yang linear dalam faktor k (O(nk)), namun tetap meningkat secara signifikan dengan bertambahnya jumlah data. Untuk Decision Trees memiliki waktu eksekusi yang lebih efisien dibandingkan SVM dan K-NN, dengan kompleksitas log-linear (O(n log n)). Untuk Neural Networks menunjukkan waktu eksekusi paling rendah, dengan pertumbuhan linear (O(n)). Ini membuat Neural Networks lebih efisien dalam hal waktu eksekusi untuk dataset yang besar.

Skalabilitas Neural Networks cenderung paling scalable untuk dataset yang besar, karena waktu komputasi meningkat paling lambat dibandingkan dengan algoritma lainnya. Decision Trees memiliki cukup scalable, tetapi dengan pertambahan data yang sangat besar, waktu eksekusinya mulai terlihat lebih signifikan. Sedangkan untuk K-NN dan SVM kurang *scalable* untuk dataset yang sangat besar, terutama SVM, yang waktu eksekusinya melonjak sangat cepat.

Pengujian berikutnya berkaitan penggunaan algoritma berdasarkan ukuran data, untuk dataset kecil hingga menengah, SVM dan K-NN mungkin masih bisa digunakan, namun perlu diperhatikan peningkatan waktu eksekusinya. Sedangkan untuk dataset yang besar, Decision Trees dan Neural Networks lebih efisien dalam hal waktu eksekusi.



Gambar 2. Hasil uji kompleksitas kombinasi ekstraksi dan klasifikasi.

4. Kesimpulan

Berdasarkan hasil pengujian dapat disimpulkan sebagai berikut, bahwa kompleksitas waktu eksekusi dari berbagai algoritma klasifikasi menunjukkan perbedaan yang signifikan ketika jumlah data bertambah. Algoritma SVM (Support Vector Machine) memiliki kompleksitas waktu eksekusi tertinggi, yang meningkat secara eksponensial seiring dengan peningkatan jumlah data, sesuai dengan kompleksitas O(n^3). Waktu eksekusi untuk SVM sangat jauh lebih besar dibandingkan dengan algoritma lain, menunjukkan bahwa SVM mungkin tidak efisien untuk digunakan pada dataset yang sangat besar. Untuk algoritma K-NN (K-Nearest Neighbors) dan Decision Trees menunjukkan peningkatan waktu eksekusi yang lebih lambat dibandingkan SVM. K-NN dengan kompleksitas O(nk) memiliki waktu eksekusi yang lebih rendah dibandingkan SVM tetapi masih signifikan lebih tinggi dibandingkan Neural Networks dan Decision Trees, terutama pada dataset yang besar. Algoritma dengan performa terbaik dalam hal efisiensi waktu adalah Neural Networks, yang menunjukkan peningkatan waktu eksekusi yang paling lambat dengan kompleksitas O(n). Meskipun waktu eksekusi meningkat seiring bertambahnya jumlah data, Neural Networks tetap lebih cepat dibandingkan algoritma lainnya. Secara keseluruhan, pemilihan algoritma untuk klasifikasi harus mempertimbangkan skala data yang akan digunakan. SVM mungkin tidak cocok untuk dataset yang sangat besar karena kebutuhan komputasi yang sangat tinggi, sementara Neural Networks dan Decision Trees menawarkan solusi yang lebih efisien dalam konteks waktu eksekusi. Dalam penerapan *IoT* (*Internet of Things*), pemilihan algoritma klasifikasi yang efisien sangat penting karena keterbatasan sumber daya komputasi dan kebutuhan untuk memproses data dalam jumlah besar secara real-time.

Daftar Pustaka

[1] R. M. Sarmento, F. F. X. Vasconcelos, P. P. R. Filho, and V. H. C. de Albuquerque, "An *IoT* platform for the analysis of brain CT images based on Parzen analysis," *Future Generation Computer Systems*, vol. 105, pp. 135–147, Apr. 2020, doi: 10.1016/j.future.2019.11.033.

[2] S. Pandiyan, M. Ashwin, R. Manikandan, K. M. Karthick Raghunath, and G. R. Anantha Raman, "Heterogeneous Internet of things organization Predictive Analysis Platform for Apple Leaf Diseases Recognition," *Comput Commun*, vol. 154, pp. 99–110, Mar. 2020, doi: 10.1016/j.comcom.2020.02.054.

[3] S. Dargan and M. Kumar, "A comprehensive survey on the biometric recognition systems based on physiological and behavioral modalities," *Expert Systems with Applications*, vol. 143. Elsevier Ltd, Apr. 01, 2020. doi: 10.1016/j.eswa.2019.113114.

[4] P. Zeng, B. Pan, K. K. R. Choo, and H. Liu, "MMDA: Multidimensional and multidirectional data aggregation for edge computing-enhanced *IoT*," *Journal of Systems Architecture*, vol. 106, Jun. 2020, doi: 10.1016/j.sysarc.2020.101713.

[5] K. Sha, T. A. Yang, W. Wei, and S. Davari, "A survey of edge computing-based designs for *IoT* security," *Digital Communications and Networks*, vol. 6, no. 2, pp. 195–202, May 2020, doi: 10.1016/j.dcan.2019.08.006.

[6] F. T. Kurniati and R. R. Huizen, "Verifikasi Dokumen Cetak Menggunakan Metode Edge Detection-Glcm Dan K-Mean Clustering," *Dinamik*, vol. 25, no. 2, pp. 85–93, 2020, doi: 10.35315/dinamik.v25i2.8188.

[7] G. Liu, J. Han, and W. Rong, "Feedback-driven loss function for small object detection," *Image Vis Comput*, vol. 111, Jul. 2021, doi: 10.1016/j.imavis.2021.104197.

[8] C. I. Ossai and N. Wickramasinghe, "GLCM and statistical features extraction technique with Extra-Tree Classifier in Macular Oedema risk diagnosis," *Biomed Signal Process Control*, vol. 73, no. November 2021, p. 103471, 2022, doi: 10.1016/j.bspc.2021.103471.

[9] S. Agrippina and A. Yusuf, "MFCC Feature Extraction and KNN Classification in ECG Signals," *2019 6th International Conference on Information Technology, Computer and Electrical Engineering (ICITACEE)*, pp. 1–5, 2019.

[10] D. A. Gustian, N. L. Rohmah, G. F. Shidik, A. Z. Fanani, R. A. Pramunendar, and Pujiono, "Classification of Troso Fabric Using SVM-RBF Multi-class Method with GLCM and PCA Feature Extraction," *Proceedings - 2019 International Seminar on Application for Technology of Information and Communication: Industry 4.0: Retrospect, Prospect, and Challenges, iSemantic 2019*, pp. 7–11, 2019, doi: 10.1109/ISEMANTIC.2019.8884329.

[11] A. Gregoriades, M. Pampaka, H. Herodotou, and E. Christodoulou, "Supporting digital content marketing and messaging through topic modelling and decision trees," *Expert Syst Appl*, vol. 184, no. August 2020, p. 115546, 2021, doi: 10.1016/j.eswa.2021.115546.

[12] Y. Yao, Z. Wang, and P. Zhou, "Privacy-preserving and energy efficient task offloading for collaborative mobile computing in *IoT*: An ADMM approach," *Comput Secur*, vol. 96, Sep. 2020, doi: 10.1016/j.cose.2020.101886.

[13] Y. li Liu, L. Huang, W. Yan, X. Wang, and R. Zhang, "Privacy in AI and the *IoT*: The privacy concerns of smart speaker users and the Personal Information Protection Law in China," *Telecomm Policy*, vol. 46, no. 7, p. 102334, 2022, doi: 10.1016/j.telpol.2022.102334.

[14] V. Zaza, M. Bisceglie, S. Valerio, and I. Giannoccaro, "The effect of complexity on the resilience and efficiency of integrated healthcare systems: the moderating role of big data analytics," *IFAC-PapersOnLine*, vol. 55, no. 10, pp. 2857–2862, 2022, doi: 10.1016/j.ifacol.2022.10.164.

[15] A. S. Nasution, A. Alvin, A. T. Siregar, and M. S. Sinaga, "KNN Algorithm for Identification of Tomato Disease Based on Image Segmentation Using Enhanced K-Means Clustering," *Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control*, vol. 4, no. 3, 2022, doi: 10.22219/kinetik.v7i3.1486.

[16] S. Salem Ghahfarrokhi and H. Khodadadi, "Human brain tumor diagnosis using the combination of the complexity measures and texture features through magnetic resonance image," *Biomed Signal Process Control*, vol. 61, p. 102025, 2020, doi: 10.1016/j.bspc.2020.102025.