KLASIFIKASI TELAPAK TANGAN MENGGUNAKAN DEEP LEARNING DAN MACHINE LEARNING

Jervis Huang[1], Teny Handhayani[2]
[1,2] Teknik Informatika, Fakultas Teknologi Informasi, Universitas Tarumanagara, Jakarta Barat, Indonesia
tenyh@fti.untar.ac.id

ABSTRAK

Tujuan penelitian ini adalah untuk membandingkan kinerja model *deep learning MobileNetV2* dan beberapa algoritma *machine learning* yaitu *K-Nearest Neighbor* (KNN), *Artificial Neural Network* (ANN), *Decision Tree* (DT) dan *Support Vector Machine* (SVM) dalam mengklasifikasi telapak tangan. Karena algoritma *machine learning* tidak dapat membaca data citra secara langsung, fitur dari gambar akan diekstraksi dengan *gray level co-occurrence matrix* (GLCM). Penelitian ini menggunakan data citra telapak tangan dari 26 responden yang berasal dari Universitas Tarumanagara yang berumur sekitar 17 tahun, dan dibagi menjadi 52 kelas (telapak tangan kanan dan kiri dianggap kelas beda). Selain itu, responden diusahakan untuk memenuhi beberapa syarat untuk data citra mereka seperti latar belakang polos dan garis-garis telapak tangan yang jelas. Data uji dan data latih diambil secara acak dengan rasio data latih:data uji 70%:30% dan 80%:20%. Setelah eksperimen, KNN menghasilkan rata-rata akurasi 87% dengan waktu latihan 0.01 detik, ANN menghasilkan rata-rata akurasi 87% dengan waktu latihan 14 detik, DT menghasilkan rata-rata akurasi 83% dengan waktu latihan 0.18 detik, SVM menghasilkan rata-rata akurasi 94% dengan waktu latihan 0.17 detik dan *MobileNetV2* menghasilkan akurasi 59% dengan waktu latihan 3336.28 detik. Selain itu, ANN mengasih tanda ketidakkonsistenan karena menghasil akurasi lebih rendah jika menggunakan data uji lebih tinggi dan MobileNetV2 mengasih tanda kekurangan data karena menghasilkan akurasi lebih rendah jika data latih lebih dikit. Dengan pengetahuan ini, para pengembang dapat memilih model atau algoritma paling cocok untuk proyek mereka, seperti sebuah sistem keamanan biometrik menggunakan telapak tangan pengguna.

Keyword : *Artificial Neural Network (ANN), Decision Tree (DT), gray level co-occurrence matrix (GLCM), K-Nearest Neighbor (KNN), MobileNetV2, Support Vector Machine (SVM).*

1. PENDAHULUAN

Telapak tangan setiap orang memiliki beberapa fitur-fitur yang berbeda seperti bentuk, tekstur, garis, dan lain-lain. Fitur tersebut bisa digunakan untuk membedakan telapak tangan antara satu orang dengan orang lain karena telapak tangan yang unik untuk setiap orang, satu orang pun tidak memiliki telapak tangan yang sama untuk tangan kiri dan tangan kanan mereka. Oleh karena itu, dengan hal-hal yang sudah dijelaskan telapak tangan bisa digunakan sebagai ciri biometrik yang bisa digunakan untuk beberapa sistem seperti sistem verifikasi untuk identitas seseorang berdasarkan dengan telapak tangan mereka.Tetapi jika seorang menganalisis telapak tangan secara manual tidak hanya kurang efisien tetapi juga sangat susah dan menggunakan waktu yang lama untuk menganalisis fitur unik dan klasifikasi telapak tangan seseorang satu-bersatu. Oleh karena itu peran tersebut akan dilakukan oleh algoritma *machine learning* dan model *deep learning*. Penggunaan *machine learning* dan *deep learning* sangat efisien karena analisis dan klasifikasi bersifat otomatis karena dilakukan oleh sebuah mesin atau perangkat (komputer, HP, laptop, dan lain-lain) tidak manual.

Klasifikasi telapak tangan dengan *algoritma machine learning* dan model *deep learning* sudah dicoba dan diuji di dalam penelitian lain menggunakan beberapa algoritma dan metode seperti mengekstraksi fitur dengan *convolutional neural network* (CNN) dan klasifikasi dengan algoritma *machine learning* SVM [1], [2]. Tetapi masih ada kekurangan penelitian yang membandingkan algoritma *machine learning* digabung dengan *gray level co-occurrence matrix* (GLCM) dan model *deep learning* dalam klasifikasi telapak tangan. Oleh karena itu, tujuan penelitian ini adalah untuk membandingkan kinerja beberapa algoritma machine learning yaitu *K-Nearest Neighbor*, *Artificial Neural Network*, *Decision Tree* dan *Support Vector machine* yang digabung dengan *Gray Level Co-occurrence Matrix* untuk mengekstraksi fitur data citra. Model deep learning *MobileNetV2* juga akan digunakan (model *deep learning* melatih dari data citra secara langsung). Karena kinerja dan hasil sebuah model atau algoritma tergantung pada data yang digunakan [3], [4], [5]. Pengetahuan yang dihasilkan dari penelitian ini dapat digunakan oleh para pengembang untuk memilih model atau algoritma paling cocok untuk proyek mereka seperti sebuah sistem keamanan biometrik menggunakan telapak tangan pengguna.

2. TINJAUAN PUSTAKA

2.1. *Gray Level Co-Occurrence Matrix* (GLCM)

Karena algoritma *machine learning* tidak dapat membaca data citra secara langsung fitur data citra akan diekstraksi menggunakan *Gray Level Co-occurrence Matrix* (GLCM). GLCM adalah metode yang digunakan untuk mengekstraksi ciri tekstur

dari data citra atau gambar seperti *energy* (jumlah piksel dalam gambar), *contrast* (intensitas kontras penghubungan piksel dan tetangganya di dalam gambar), *correlation* (korelasi piksel dengan tetangganya), dan *homogeneity* (kesamaan piksel). Rumus untuk ekstraksi dari fitur tersebut bisa dilihatkan dari rumus 1-4 [9], [10].

$$Energy = \sum_{i,j=0}^{N-1} \left(P_{ij}\right)^2 \quad (1)$$

$$Contrast = \sum_{i,j=0}^{N-1} P_{ij}(i-j)^2 \quad (2)$$

$$Correlation = \sum_{i,j=0}^{N-1} \frac{(i-\mu)(j-\mu)}{\sigma^2} \quad (3)$$

$$Homogeneity = \sum_{i,j=0}^{N-1} \frac{P_{ij}}{1+(i-j)^2} \quad (4)$$

Di mana P_{ij} adalah elemen gambar, μ adalah nilai rata-rata matriks GLCM, σ adalah perbedaan piksel, n adalah jumlah tingkat warna di dalam gambar.

2.2. Metrik Evaluasi

Kinerja algoritma *machine learning* dan model *deep learning* yang bertujuan untuk klasifikasi bisa diperiksa dengan menghitung beberapa metrik seperti akurasi, presisi, *recall* dan *f1-score* [11], [12]:

a. Akurasi (prediksi yang benar)
$$accuracy = \frac{TP+TN}{P+N} \quad (5)$$
b. Presisi (kemampuan memprediksi positif benar)
$$precision = \frac{TP}{TP+FP} \quad (6)$$
c. Recall (kemampuan mendeteksi positif)
$$recall = \frac{TP}{P} \quad (7)$$
d. *F1-score* (rata-rata harmonik dari presisi dan *recall*)
$$f1\ score = 2 * \frac{precision * recall}{precision + recall} \quad (8)$$

Dimana:

a. TP adalah *True Positive* (prediksi positif yang benar)
b. TN adalah *True Negative* (prediksi negatif yang benar)
c. FP adalah *False Positive* (prediksi positif yang salah)
d. FN adalah *False Negative* (prediksi negatif yang salah)
e. P adalah jumlah positif
f. N adalah jumlah negatif

Semua metrik terdapat dalam bentuk persentase berarti 1 sama dengan 100%. Selain metrik yang sudah dijelaskan, waktu latihan yaitu waktu yang digunakan oleh algoritma untuk mempelajari data akan dicatat juga.

2.3. *Artificial Neural Network* (ANN)

Artificial neural network (ANN) adalah sebuah algoritma *machine learning* yang dikembangkan oleh Warren McCullough dan Walter Pitts yang terdiri dari beberapa lapisan perceptron dan perceptron adalah sebuah algoritma yang digunakan oleh *supervised learning* dan *binary classifier*. *Binary classifier* adalah sebuah fungsi yang dapat mengklasifikasikan suatu input (data) yang direpresentasikan dengan vektor [13], [14], [15]. ANN memiliki lapisan *node* yang merepresentasikan *input*, *hidden*, dan *output*. *Node* tersebut dibuat oleh ANN ketika diberi data uji. Cara ANN prediksi data baru adalah dengan memilih *output node* yang paling sesuai dengan *input node*.

2.4. *K-Nearest Neighbor* (KNN)

K-Nearest Neighbor (KNN) adalah sebuah algoritma *machine learning* yang dikembangkan oleh Evelyn Fix dan Joseph Hodges di tahun 1951. Algoritma ini menyimpan data secara langsung dan memprediksi data baru dengan mencari K data yang paling mirip atau dekat dengan data baru[16], [17]. Jumlah K bersifat integer dan bisa diganti. Selain itu K juga biasanya bersifat ganjil karena jika K adalah genap seperti 2 dan dua data yang paling dekat dengan data baru adalah positif dan negatif algoritma tidak bisa prediksi karena berdua jawaban bisa dianggap benar.

2.5. *Decision Tree* (DT)

Decision Tree (DT) adalah algoritma *machine learning* yang dikembangkan oleh John Sonquist dan James Morgan. Algoritma ini mulai dari suatu *node* akar (*node* paling pertama) dan ketika jika data baru ditambah sebuah *node* baru akan ditambah tergantung pada kriteria data yang ditambah. DT prediksi data baru dengan melihat dimana data baru tersebut diletakkan sesuai dengan kondisi atau kriteria yang sudah dibuatkan [18], [19].

2.6. *Support Vector Machine* (SVM)

Support Vector Machine (SVM) adalah algoritma *machine learning* yang dikembangkan oleh Vladimir N. Vapnik. Algoritma ini menerima *input* dari sebuah vektor lalu vektor tersebut dipetakan ke dalam sebuah *high dimensional feature space* (HDFS) menggunakan sesuatu bernama *nonlinear mapping* dan dari bantuan dari sesuatu bernama *kernel* jika data tersebut tidak dapat dipisahkan secara linier. Setelah itu, sebuah pemisah yang optimal akan dibuatkan di ruangan tersebut. SVM prediksi data baru dengan melihat sisi mana data baru diletakkan [20], [21], [22].

2.7. *Deep Learning* (DL)

Deep Learning (DL) adalah subkategori *machine learning*. *Deep learning* adalah sebuah kelas yang terdiri dari beberapa lapisan antara *input* dan *output* [23], [24], [25], [26]. Oleh karena itu DL juga bisa dipanggilkan *deep neural network* (DNN) dan mirip dengan ANN tetapi perbedaan utama antara mereka berdua adalah ANN perlu fitur untuk diekstraksi sebelum melatih atau menguji, tetapi DL dapat mengenali fitur tanpa ekstraksi. Selain itu dibanding dengan ANN, DL mempunyai layar lebih banyak dari ANN oleh karena itu, menggunakan waktu dan sumber daya lebih banyak [27].

2.8. *MobileNetV2*

MobileNetV2 adalah model DL yang dikembangkan oleh google pada tahun 2019 yang dibuat untuk pengenalan gambar. Model ini dikembangkan untuk lingkungan atau perangkat yang mempunyai sumber daya yang terbatas seperti HP. Model ini menggunakan *inverted residual layers* dan *linear bottleneck*. *Inverted residual layers* adalah metode yang digunakan untuk mengurangi *node*. *Linear bottleneck* adalah metode yang digunakan untuk mengganti fungsi aktivasi yang digunakan karena fungsi aktivasi yang biasa bermasalah karena jumlah *node* yang dikurangi dari *inverted residual layers* [28], [29], [30], [31].

2.9. *Random Sampling*

Random sampling adalah salah satu metode pengujian yang digunakan untuk mengevaluasi kinerja sebuah algoritma machine learning. Metode pengujian ini dimulai dengan memilih beberapa persen data yang akan digunakan sebagai data latih dan data pengujian secara acak [32].

2.10. *Pre-processing*

Pre-Processing adalah proses yang digunakan untuk memastikan dan menyiapkan bahwa data untuk *Machine Learning* serta *Deep Learning* sebaik mungkin. *Pre-processing* mencakupi beberapa proses untuk memastikan kualitas data sebaik mungkin [33]. Proses yang digunakan untuk penelitian ini adalah:

a. Pemeriksaan duplikat, jika ketemu duplikat akan dihapus.
b. Standarisasi, standarisasi adalah proses dimana data ditransformasikan ke sebuah format standar [34]. Data yang telah ditransformasikan membuat algoritma belajar dari data lebih efisien dan menghasilkan hasil yang lebih tinggi. Rumus standarisasi bisa dilihatkan pada rumus 9 (proses ini khusus untuk algoritma *machine learning* karena fitur GLCM terdapat di dalam bentuk numerik).

$$x\ baru = \frac{x\ lama - mean}{sd} \qquad (9)$$

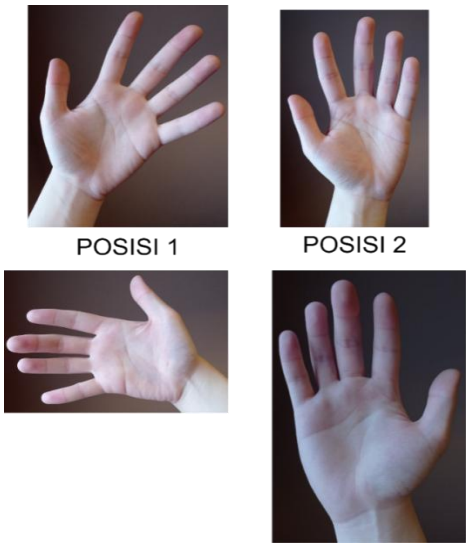
Di mana *mean* adalah rata-rata data, sd adalah deviasi standar dan x adalah data yang sedang dihitung.

3. METODE PENELITIAN

3.1. Data

Data yang digunakan untuk penelitian ini adalah data citra telapak tangan yang diperoleh dari responden, responden tersebut terdiri mahasiswa Universitas Tarumanagara yang berusia sekitar 17 sampai 30 tahun. Setiap responden memberikan sebanyak 100 data citra telapak tangan. Sekitar 25 data citra tersebut terdiri dari telapak tangan kiri di posisi 1, telapak tangan kiri di posisi 2, telapak tangan kanan di posisi 1, telapak tangan kanan di posisi 2 (posisi dan contoh data citra bisa dilihat di gambar 1). Jumlah responden yang terdapat untuk penelitian ini adalah 30 yang menjadi 60 karena telapak tangan kanan dan kiri dianggap kelas beda. Selain itu responden juga diusahakan untuk memenuhi dua syarat agar data sebaik mungkin yaitu:

a. Garis-garis di telapak tangan dan jari dapat dilihat dengan jelas
b. Latar belakang (*background*) diusahakan bersih dari objek pengganggu (*background* polos)



Gambar 1. Contoh data citra telapak tangan (atas adalah tangan kiri dan bawah adalah tangan kanan)

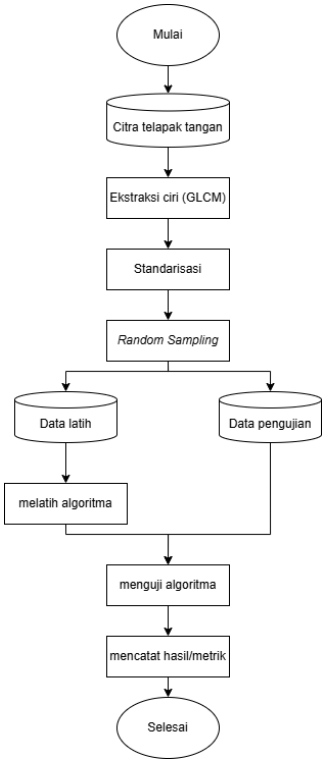
3.2. Metode Evaluasi

Eksperimen akan dijalankan di dalam lingkungan bahasa pemrograman *python* versi 3.11.13. Untuk algoritma KNN dan ANN, eksperimen akan menggunakan *library* yang tersedia di dalam *scikit-learn*. Untuk model *MobileNetV2*, eksperimen akan menggunakan *library* yang tersedia di dalam *tensorflow* dan *keras*. Selain itu, ada *library* yang digunakan seperti *numpy* untuk mempermudahkan perhitungan, *matplotlib.pyplot* untuk menggambarkan grafik, dan lain-lain.

3.3. Proses Pengujian

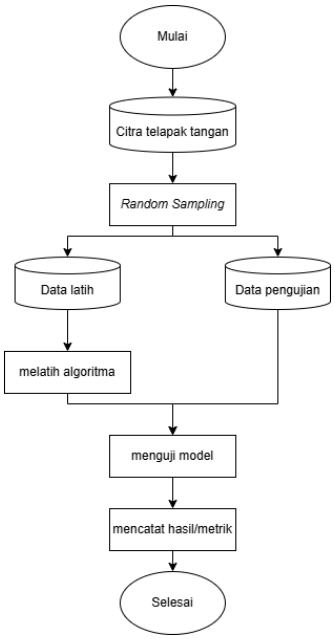
Sebelum eksperimen mulai proses *pre-processing* akan dijalankan dulu. Ketika pemeriksaan data telah ditemukan beberapa data citra yang menggunakan format *file .heic* yang tidak bisa dibaca oleh *python* oleh karena itu data tersebut akan dihapus, setelah sudah menghapus sisa responden adalah 26 dengan 52 kelas. Setelah itu fitur data citra akan diekstraksi dengan GLCM. Setelah ekstraksi fitur selesai, fitur tersebut akan distandarisasi.

Jumlah data uji dan data latih akan dipilihkan oleh random sampling dengan rasio data latih:data uji 70%:30% dan 80%:20% untuk melihat apakah kinerja meningkat atau berkurang di berdua skenario. Sebelum pengujian mulai algoritma KNN akan diuji untuk mencari nilai K paling tepat (menghasil akurasi paling tinggi). Nilai K yang akan diuji adalah semua nomor ganjil dari 3 sampai 101.



Gambar 2. Proses pengujian untuk algoritma Machine Learning

Setelah itu eksperimen akan dimulai, semua algoritma dan model akan diujikan 5 kali. setelah semua eksperimen selesai rata-rata metrik dan waktu latihan akan dihitung dan dicatat. *Flowchart* untuk proses pengujian bisa dilihat dari gambar 2 dan 3 di dalam gambar tersebut data citra telapak tangan sudah melewati proses pre-processing jadi hanya sisa standarisasi untuk algoritma machine learning. Dan setelah eksperimen selesai, karena *library* yang digunakan untuk deep learning terdapat fitur untuk melihat proses latihan grafik dari proses tersebut juga akan ditampilkan (hanya 1 dari 5 eksperimen untuk setiap skenario akan ditampilkan).



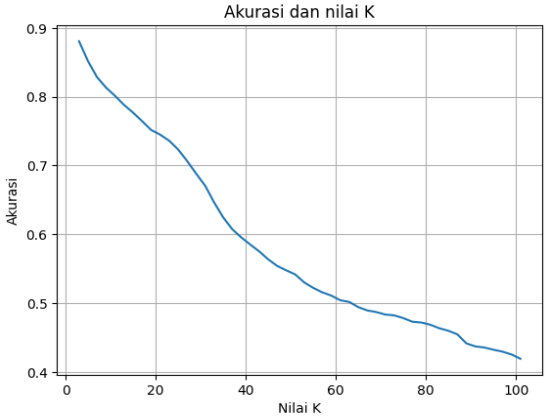
Gambar 3. Proses pengujian untuk MobileNetV2

3.4. Persiapan Eksperimen

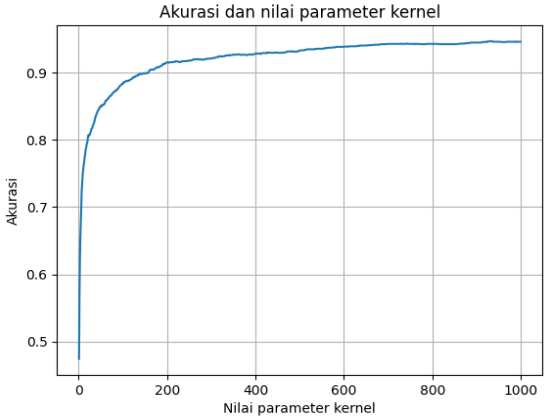
Sebelum eksperimen mulai nilai parameter untuk setiap algoritma dan model akan ditentukan sesuai berikut:

a. KNN akan menggunakan nilai K 3 (grafik untuk penemuan nilai k paling tepat bisa dilihatkan di gambar 4).
b. SVM akan menggunakan nilai parameter *kernel* 928 (grafik untuk penemuan nilai parameter *kernel* paling tepat bisa dilihatkan di gambar 5).
c. ANN akan menggunakan *max_iter* 1000 karena dalam pengujian, *max_iter* 1000 menghasilkan hasil yang cukup tinggi tanpa menggunakan waktu yang terlalu banyak (*max_iter* lebih tinggi menyebabkan hasil lebih bagus tetapi menggunakan waktu latihan lebih banyak)
d. DT akan menggunakan *max_depth* 1000 karena dalam pengujian, *max_iter* 1000 menghasilkan hasil yang cukup tinggi tanpa menggunakan waktu yang terlalu banyak (*max_iter* lebih tinggi menyebabkan hasil lebih bagus tetapi menggunakan waktu latihan lebih banyak)
e. *MobileNetV2* menggunakan model yang sudah terlatih dari *imagenet* yang berspesialisasi untuk pengenalan gambar, fungsi aktivasi yang digunakan adalah *relu* dan *softmax* dan pengoptimal (*optimizer*) yang digunakan adalah *Adam.*
f. *Epoch* yang digunakan untuk *MobileNetV2* adalah 10 karena di dalam pengujian *MobileNetV2* sudah memiliki waktu latihan yang jauh lebih lama dibanding algoritma ML oleh karena itu, nomor *epoch* yang rendah akan digunakan (*epoch* meningkatkan hasil tetapi menggunakan waktu latihan yang banyak).
g. *Steps_per_epoch* yang digunakan untuk *MobileNetV2* adalah 50 karena di dalam

pengujian *MobileNetV2* sudah memiliki waktu latihan yang jauh lebih lama dibanding algoritma ML oleh karena itu, nomor *steps_per_epoch* yang rendah akan digunakan (steps_per_*epoch* meningkatkan hasil tetapi menggunakan waktu latihan yang banyak).



Gambar 4. Grafik nilai K untuk KNN

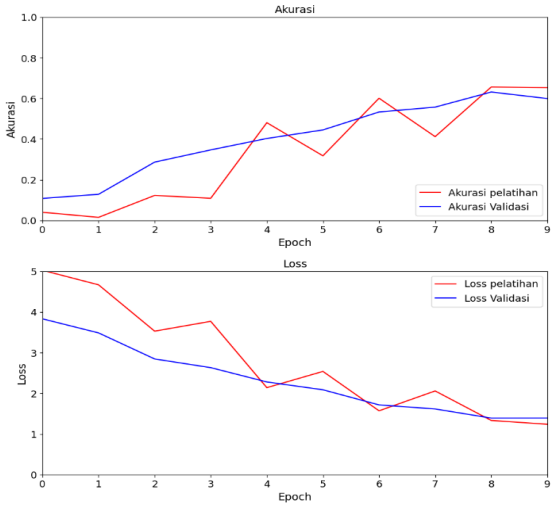


Gambar 5. Grafik nilai parameter kernel untuk SVM

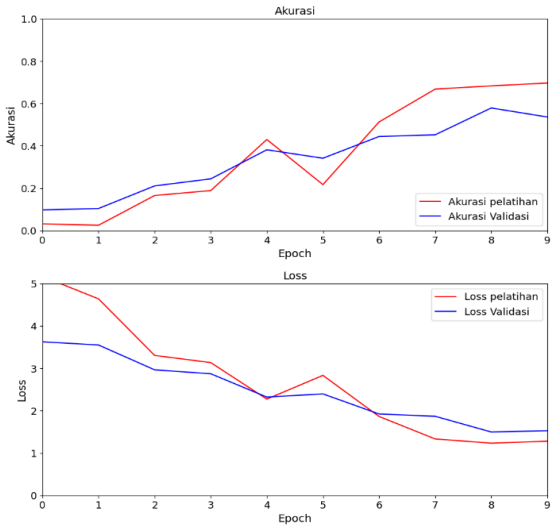
4. HASIL DAN PEMBAHASAN

4.1. Hasil Eksperimen

Untuk penelitian ini hasil eksperimen akan ditulis di dalam format tabel dan grafik. Untuk grafik *deep learning*, *epoch* adalah beberapa kali model telah pelajari data, dan *loss* adalah perbedaan predkisi yang salah dari kelas sebenarnya grafik tersebut bisa dilihat pada gambar 6 dan 7. Hasil eksperimen dapat dilihat di dalam Tabel 1 dan 2, dan grafik hasil untuk eksperimen bisa dilihat di dalam gambar 8 dan 9.



Gambar 6. Grafik proses latihan model MobileNetV2 dengan rasio data latih:data uji 80%:20%



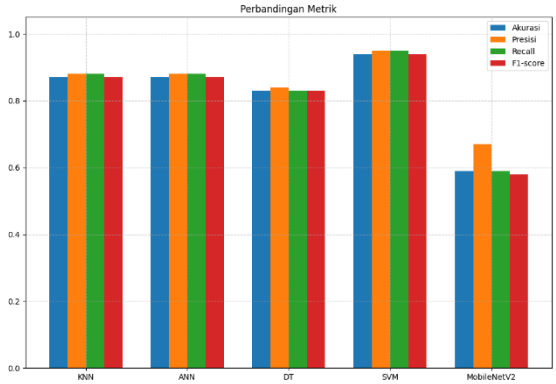
Gambar 7. Grafik proses latihan model MobileNetV2 dengan rasio data latih:data uji 70%:30%

Tabel 1. Hasil eksperimen sesuai skenario dan algoritma/model

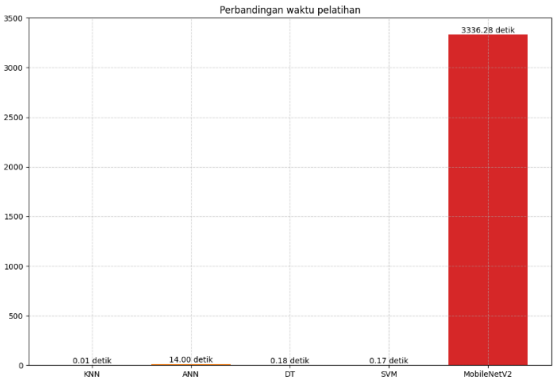
Algoritma/model rasio data latih: data uji	Akurasi	Presisi	*Recall*	*F1-score*	Waktu Latihan (detik)
KNN 80%:20%	0.87	0.88	0.87	0.87	0.01
KNN 70%:30%	0.87	0.88	0.88	0.87	0.01
ANN 80%:20%	0.82	0.83	0.83	0.81	13.21
ANN 70%:30%	0.92	0.93	0.93	0.92	14.79
DT 80%:20%	0.83	0.84	0.84	0.83	0.18
DT 70%:30%	0.82	0.83	0.82	0.82	0.17
SVM 80%:20%	0.94	0.94	0.95	0.94	0.26
SVM 70%:30%	0.94	0.95	0.94	0.94	0.07
Mobile NetV2 80%:20%	0.63	0.70	0.63	0.61	2903.60
Mobile NetV2 70%:30%	0.55	0.63	0.55	0.54	3768.95

Tabel 2. Hasil eksperimen keseluruhan (rata-rata dari berdua skenario)

Algoritma/model dan rasio data latih: data uji	Akurasi	Presisi	*Recall*	*F1-score*	Waktu Latihan (detik)
KNN	0.87	0.88	0.88	0.87	0.01
ANN	0.87	0.88	0.88	0.87	14.00
DT	0.83	0.84	0.83	0.83	0.18
SVM	0.94	0.95	0.95	0.94	0.17
Mobile NetV2	0.59	0.67	0.59	0.58	3336.28



Gambar 8. Grafik perbandingan metrik evaluasi



Gambar 9. Grafik perbandingan waktu latihan

4.2. Analisis Hasil Eksperimen

MobileNetV2 kurang tepat untuk data citra telapak tangan karena metrik yang lumayan rendah dan model ini memiliki waktu latihan yang paling lama dari semua algoritma. Tetapi, ada kemungkinan metrik bisa lebih tinggi jika *epoch* atau *steps_per_epoch* ditingkatkan karena di dalam gambar 6 dan 7 akurasi meningkat dan *loss* menurun. Tetapi meningkatkan berdua parameter tersebut juga akan meningkat waktu latihannya. Selain meningkat jumlah epoch, kemungkinan nilai akurasi yang rendah bisa disebabkan oleh kekurangan data yang kemudian mengakibatkan *underfitting*. Model ini menghasilkan akurasi lebih tinggi jika diberikan lebih banyak data uji dan menurun jika diberikan data uji lebih dikit. Selain itu, model ini memiliki presisi yang lebih tinggi dari *recall* yang berarti model ini tidak prediksi banyak kasus yang positif tetapi ketika model ini prediksi sesuatu kasus sebagai positif kasus tersebut ada kemungkinan besar beneran positif.

ANN mempunyai perbedaan metrik sebesar ~10% di berdua skenario *random sampling* ini menunjukkan bahwa kinerja ANN lebih rendah jika menggunakan data latih yang lebih banyak (menunjukkan tanda-tanda *overfitting*). Selain itu, ANN menghasilkan rata-rata metrik yang cukup tinggi (sama dengan KNN) dan waktu latihan kedua paling lambat. Akurasi ANN juga dapat ditingkatkan jika jumlah parameter *max_iter* ditambahkan tetapi waktu latihan juga menjadi lebih lambat.

KNN mempunyai metrik yang cukup tinggi dan konsisten karena metrik yang dihasilkan mirip di dalam berdua skenario *random sampling* ini kemungkinan karena data GLCM yang distandarisasi dapat di. Selain itu, KNN juga menghasilkan waktu latihan paling cepat.

DT mempunyai metrik yang cukup tinggi dan konsisten karena metrik yang dihasilkan mirip di dalam berdua skenario *random sampling*. Selain itu, DT juga menghasilkan waktu latihan ketiga paling cepat.

SVM mempunyai metrik yang paling tinggi dan konsisten karena metrik yang dihasilkan mirip di dalam berdua skenario *random sampling* kecuali waktu latihan yang lebih cepat jika data latihan lebih dikit. Selain itu, SVM juga menghasilkan waktu latihan kedua paling cepat.

Hasil yang tinggi untuk KNN, DT dan SVM kemungkinan bisa disebabkan oleh data GLCM yang distandarisasi dapat diprediksi oleh algoritma-algoritma tersebut dengan baik.

5. KESIMPULAN DAN SARAN

Kesimpulan dari penelitian ini adalah algoritma SVM adalah algoritma paling tepat jika pengguna atau pengembang ingin sebuah algoritma yang menghasilkan akurasi yang tinggi dan KNN jika ingin algoritma yang menghasilkan akurasi yang cukup tinggi dan waktu latihan yang cepat. *MobileNetV2* juga berpotensi untuk menghasilkan hasil yang lebih tinggi jika lebih banyak data uji digunakan dan jika parameter seperti *epoch* ditingkatkan. Oleh karena itu, algoritma *machine learning* SVM dan KNN dapat digunakan untuk mengembangkan aplikasi sistem keamanan menggunakan telapak tangan. Untuk urutan model dan algoritma dari paling tidak cocok hingga yang paling cocok untuk dataset yang digunakan adalah *MobileNetV2*, ANN (karena tidak konsisten dengan data yang lebih banyak), DT, KNN dan SVM.

Saran penelitian berikutnya, algoritma *machine learning* yang lain bisa diujikan untuk mencari algoritma paling tepat untuk mengklasifikasi telapak tangan. Model *deep learning* lain juga dapat diujikan untuk mencari model paling tepat untuk klasifikasi telapak tangan dan model yang digunakan untuk penelitian ini bisa dioptimalisasikan seperti mengganti beberapa parameter atau menambah lebih banyak data uji. Hasil dari penelitian ini dapat digunakan oleh para pengembang untuk memilih model atau algoritma paling cocok untuk proyek mereka seperti sebuah sistem keamanan biometrik menggunakan telapak tangan pengguna

DAFTAR PUSTAKA

[1] N. Amrouni, A. Benzaoui, and A. Zeroual, "Palmprint Recognition: extensive exploration of databases, methodologies, comparative assessment, and future directions," *Applied Sciences*, vol. 14, no. 1, p. 153, 2023.

[2] S. A. M. Al-Taie and B. I. Khaleel, "Palm print recognition using intelligent techniques: A review," *Jurnal Ilmiah Teknik Elektro Komputer dan Informatika*, vol. 9, no. 1, pp. 156–164, 2023.

[3] J. J. Khanam and S. Y. Foo, "A comparison of machine learning algorithms for diabetes prediction," *Ict Express*, vol. 7, no. 4, pp. 432–439, 2021.

[4] N. Thapa, Z. Liu, D. B. Kc, B. Gokaraju, and K. Roy, "Comparison of machine learning and deep learning models for network intrusion detection systems," *Future Internet*, vol. 12, no. 10, p. 167, 2020.

[5] Z. Liu *et al.*, "Accuracy analyses and model comparison of machine learning adopted in building energy consumption prediction," *Energy Exploration & Exploitation*, vol. 37, no. 4, pp. 1426–1451, 2019.

[6] E. Sullivan, "Understanding from machine learning models," *Br J Philos Sci*, 2022.

[7] J. M. Aguiar-Pérez, M. A. Pérez-Juárez, M. Alonso-Felipe, J. Del-Pozo-Velázquez, S. Rozada-Raneros, and M. Barrio-Conde, "Understanding machine learning concepts," in *Encyclopedia of data science and machine learning*, IGI Global, 2023, pp. 1007–1022.

[8] J. Lehr, J. Philipps, V. N. Hoang, D. von Wrangel, and J. Krüger, "Supervised learning vs. unsupervised learning: A comparison for optical inspection applications in quality control," in *Iop conference series: Materials science and engineering*, IOP Publishing, 2021, p. 012049.

[9] D. Kumar, "Feature extraction and selection of kidney ultrasound images using GLCM and PCA," *Procedia Comput Sci*, vol. 167, pp. 1722–1731, 2020.

[10] S. Barburiceanu, R. Terebes, and S. Meza, "3D texture feature extraction and classification using GLCM and LBP-based descriptors," *Applied Sciences*, vol. 11, no. 5, p. 2332, 2021.

[11] O. Rainio, J. Teuho, and R. Klén, "Evaluation metrics and statistical tests for machine learning," *Sci Rep*, vol. 14, no. 1, p. 6086, 2024.

[12] B. J. Erickson and F. Kitamura, "Magician's corner: 9. Performance metrics for machine learning models," 2021, *Radiological Society of North America*.

[13] P. P. Angelov, E. A. Soares, R. Jiang, N. I. Arnold, and P. M. Atkinson, "Explainable artificial intelligence: an analytical review," *Wiley Interdiscip Rev Data Min Knowl Discov*, vol. 11, no. 5, p. e1424, 2021.

[14] M. G. M. Abdolrasol *et al.*, "Artificial neural networks based optimization techniques: A review," *Electronics (Basel)*, vol. 10, no. 21, p. 2689, 2021.

[15] A. Mollalo, K. M. Rivera, and B. Vahedi, "Artificial neural network modeling of novel coronavirus (COVID-19) incidence rates across the continental United States," *Int J Environ Res Public Health*, vol. 17, no. 12, p. 4204, 2020.

[16] S. Zhang, "Challenges in KNN classification," *IEEE Trans Knowl Data Eng*, vol. 34, no. 10, pp. 4663–4675, 2021.

[17] N. F. Rajani, B. Krause, W. Yin, T. Niu, R. Socher, and C. Xiong, "Explaining and improving model behavior with k nearest neighbor representations," *arXiv preprint arXiv:2010.09030*, 2020.

[18] Z. Azam, M. M. Islam, and M. N. Huda, "Comparative analysis of intrusion detection systems and machine learning-based model analysis through decision tree," *IEEE Access*, vol. 11, pp. 80348–80391, 2023.

[19] C. S. Lee, P. Y. S. Cheang, and M. Moslehpour, "Predictive analytics in business analytics: decision tree," *Advances in Decision Sciences*, vol. 26, no. 1, pp. 1–29, 2022.

[20] J. M. Ashfaque and A. Iqbal, "Introduction to support vector machines and kernel methods," *no. April*, pp. 1–9, 2019.

[21] S. Suthaharan, "Support vector machine," in *Machine learning models and algorithms for big data classification: thinking with examples for effective learning*, Springer, 2016, pp. 207–235.

[22] D. Valkenborg, A.-J. Rousseau, M. Geubbelmans, and T. Burzykowski, "Support vector machines," *American Journal of Orthodontics and Dentofacial Orthopedics*, vol. 164, no. 5, pp. 754–757, 2023.

[23] P. L. Bartlett, A. Montanari, and A. Rakhlin, "Deep learning: a statistical viewpoint," *Acta numerica*, vol. 30, pp. 87–201, 2021.

[24] Y. Bengio, Y. Lecun, and G. Hinton, "Deep learning for AI," *Commun ACM*, vol. 64, no. 7, pp. 58–65, 2021.
[25] N. Sharma, R. Sharma, and N. Jindal, "Machine learning and deep learning applications-a vision," *Global Transitions Proceedings*, vol. 2, no. 1, pp. 24–28, 2021.
[26] K. Sharifani and M. Amini, "Machine learning and deep learning: A review of methods and applications," *World Information Technology and Engineering Journal*, vol. 10, no. 07, pp. 3897–3904, 2023.
[27] N. C. Thompson, K. Greenewald, K. Lee, and G. F. Manso, "The computational limits of deep learning," *arXiv preprint arXiv:2007.05558*, vol. 10, 2020.
[28] K. Sabanci, "Benchmarking of CNN models and mobilenet-BiLSTM approach to classification of tomato seed cultivars," *Sustainability*, vol. 15, no. 5, p. 4443, 2023.
[29] M. Rybczak and K. Kozakiewicz, "Deep machine learning of MobileNet, Efficient, and Inception models," *Algorithms*, vol. 17, no. 3, p. 96, 2024.
[30] Y. Gulzar, "Fruit image classification model based on MobileNetV2 with deep transfer learning technique," *Sustainability*, vol. 15, no. 3, p. 1906, 2023.
[31] F. Masykur, M. B. Setyawan, and K. Winangun, "Epoch optimization on rice leaf image classification using Convolutional Neural Network (CNN) mobilenet," *CESS (Journal of Computer Engineering, System and Science)*, vol. 7, no. 2, pp. 581–591, 2022.
[32] C. A. Ramezan, T. A. Warner, and A. E. Maxwell, "Evaluation of sampling and cross-validation tuning strategies for regional-scale machine learning classification," *Remote Sens (Basel)*, vol. 11, no. 2, p. 185, 2019.
[33] K. Maharana, S. Mondal, and B. Nemade, "A review: Data pre-processing and data augmentation techniques," *Global Transitions Proceedings*, vol. 3, no. 1, pp. 91–99, 2022.
[34] P. J. M. Ali, R. H. Faraj, E. Koya, P. J. M. Ali, and R. H. Faraj, "Data normalization and standardization: a technical report," *Mach Learn Tech Rep*, vol. 1, no. 1, pp. 1–6, 2014.