

Verifikasi Tanda Tangan Menggunakan Model *BiT-M-R50x1* dan *Support Vector Machine* (SVM)

Yulian Safarudin¹, Regina Lionnie²

Program Studi Magister Teknik Elektro,
Fakultas Teknik,
Universitas Mercu Buana, Jakarta

¹55422120001@student.mercubuana.ac.id, ²uregina.lionnie@mercubuana.ac.id

Ringkasan

Verifikasi tanda tangan adalah aspek penting dalam administrasi individu dan lembaga keuangan, terutama untuk mencegah pemalsuan yang dapat menimbulkan dampak hukum serius. Berdasarkan data Direktori Putusan Mahkamah Agung, tercatat 471 kasus pemalsuan tanda tangan selama 2021–2023, sehingga diperlukan metode verifikasi yang akurat. Penelitian ini menggabungkan model *BiT-M-R50x1* dan *Support Vector Machine* (SVM). *Dataset* yang digunakan dalam pengujian berasal dari *Kaggle* dengan 2.149 gambar tanda tangan asli dan palsu. Pengujian dilakukan dengan menggunakan model *BiT-M-R50x1* dengan *preprocessing noise removal, skeletonization, Region of Interest (ROI), merging of images, ImageDataGenerator* dan ekstraksi fitur *Grey Level Co-occurrence Matrix (GLCM)* dan *Red Green Yellow (RGY)*. Hasil penelitian menunjukkan bahwa *preprocessing* tambahan seperti *noise removal, skeletonization, region of interest (ROI), merging of images*, serta ekstraksi fitur *GLCM* dan *RGY* menghasilkan performa lebih rendah dibandingkan metode tanpa *preprocessing* dan ekstraksi fitur. Kombinasi *BiT-M-R50x1* dan SVM dengan *kernel linear* memberikan hasil terbaik pada *validation set* (*accuracy* 0,9970; *precision* 0,9935; *recall* 1,0000; *F1 score* 0,9967) dan *test set* (*accuracy, precision, recall, dan F1 score* 1,0000), baik dengan maupun tanpa *preprocessing Image Data Generator*. Pengujian model tanpa *preprocessing* dan ekstraksi fitur pada *dataset* yang dirusak dengan *blur* dan *noise* dengan jumlah kerusakan *dataset* 25%, 50% dan 75% dari seluruh jumlah *dataset* menunjukkan penurunan performa, tetapi *kernel linear* tetap memberikan hasil terbaik di semua tingkat kerusakan. Penelitian ini menyimpulkan bahwa *BiT-M-R50x1* dan SVM dengan *kernel linear* adalah kombinasi optimal untuk verifikasi tanda tangan, sementara *preprocessing* dan ekstraksi fitur tambahan tidak selalu meningkatkan performa.

Kata kunci: Verifikasi tanda tangan, *BiT-M-R50x1*, *Support Vector Machine* (SVM)

1. Pendahuluan

Tanda tangan tulisan tangan merupakan salah satu atribut pribadi yang paling banyak digunakan, dianggap sebagai sarana sah otentikasi administratif individu dan lembaga keuangan. Tanda tangan setiap individu dapat didefinisikan sebagai kombinasi simbol dan coretan yang pada dasarnya merupakan atribut individu dan mewakili gaya penulisan unik. Tanda tangan digunakan untuk mengidentifikasi seseorang dan keaslian dokumen [1].

Pemalsuan tanda tangan masuk dalam bentuk pemalsuan surat yang dapat dijerat dengan Pasal 263 ayat (1) KUHP. Pelakunya diancam dengan pidana penjara selama enam

tahun. Lebih jelasnya, Pasal 263 ayat (1) KUHP menyatakan bahwa barangsiapa membuat secara tidak benar atau memalsu surat yang dapat menimbulkan sesuatu hak, perikatan atau pembebasan utang, atau yang diperuntukkan sebagai bukti dari sesuatu hal, dengan maksud untuk memakai atau menyuruh orang lain pakai surat tersebut seolah-olah isinya benar dan tidak dipalsu, diancam, jika pemakaian tersebut dapat menimbulkan kerugian, karena pemalsuan surat, dengan pidana penjara paling lama enam tahun [2].

Ada 471 kasus yang melapor terkait pemalsuan tanda tangan berdasarkan data Putusan yang diupload oleh Direktori Putusan Mahkamah Agung (MA), sejak tahun 2021 hingga 2023 [3].

Pengolahan citra digital adalah ilmu yang mempelajari hal-hal berkaitan dengan perbaikan kualitas terhadap suatu gambar (meningkatkan kontras, perubahan warna, restorasi citra), transformasi gambar (translasi, rotasi transformasi, skala, geometrik), melakukan pemilihan citra ciri (*feature images*) yang optimal untuk tujuan analisis, melakukan penyimpanan data yang sebelumnya dilakukan reduksi dan kompresi, transmisi data, dan waktu proses data.

Penelitian yang dilakukan oleh Tanvir, Kazi dengan judul *Signature Verification System: Using Big Transfer (BiT-M-R50x1) for Accurate Authentication*, menunjukkan keandalan yang luar biasa dengan akurasi pengujian sebesar 99,40%. Lebih lanjut dalam penelitian lain yang berjudul *Offline Signature Verification: An Application of GLCM Features in Machine Learning* oleh Singh, Prashant dkk menunjukkan bahwa model *Support Vector Machine* (SVM) menunjukkan kinerja yang lebih baik dalam klasifikasi gambar dibandingkan dengan model *Convolutional Neural Network* (CNN) [4][5].

Pada penelitian ini penulis akan menggunakan SVM yang dikombinasikan dengan *BiT-M-R50x1* untuk melakukan proses verifikasi tanda tangan. Diharapkan dengan penggunaan SVM dan *BiT-M-R50x1* secara bersamaan hasil dari verifikasi tanda tangan akan lebih maksimal.

2. Metode Penelitian

Pada bagian ini akan dijelaskan *dataset* yang digunakan pada penelitian metode pengolahan data, metode analisa data dan desain penelitian. Penjelasan pada setiap subbab dirancang untuk memberikan gambaran mengenai tahapan penelitian yang dilakukan.

2.1. Data Penelitian

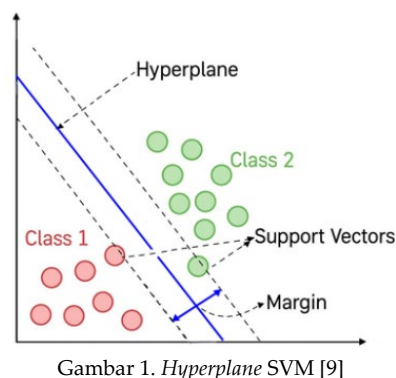
Dataset yang digunakan oleh penulis sama dengan *dataset* yang digunakan oleh K. Tanvir dalam penelitian yang berjudul "*Signature Verification System: Using Big Transfer (BiT-M-R50x1) for Accurate Authentication*" yaitu *dataset* yang diperoleh dari situs [www.kaggle.com](https://www.kaggle.com/robinreni/signature-verification-dataset) yaitu [robinreni/signature-verification-dataset](https://www.kaggle.com/robinreni/signature-verification-dataset). *Dataset* ini memiliki 2149 gambar dimana gambar tersebut dibagi menjadi 2, yaitu asli dan palsu. Untuk folder dengan gambar tanda tangan asli, penamaan foldernya dengan angka, sedangkan untuk folder dengan gambar tanda tangan palsu foldernya angka seperti penamaan folder gambar tanda tangan asli dengan imbuhan "*_forg*". Pada *dataset* ini jumlah gambar yang digunakan untuk pengujian ada 500 gambar, dengan jumlah gambar tanda tangan asli 252 gambar dan gambar tanda tangan palsu 248 gambar. Sedangkan jumlah gambar yang digunakan untuk training ada 1649 gambar dengan jumlah gambar tanda tangan asli 887 gambar dan jumlah gambar tanda tangan palsu 762 gambar [4].

2.2. BiT-M-R50x1

Model *BiT-M-R50x1* adalah sebuah model *Convolutional Neural Network* (CNN) berperforma tinggi yang didasarkan pada arsitektur *ResNet*. Model ini dirancang untuk menangani berbagai tugas seperti *image classification*, *object detection*, dan *image segmentation* dengan memanfaatkan *transfer learning* pada model yang telah di-*pre-trained* [6]. *Transfer learning* adalah teknik yang memanfaatkan model pra-latih (model yang telah dilatih menggunakan *dataset* tertentu) untuk menyelesaikan masalah lain yang memiliki kemiripan. Teknik ini dimulai dengan menggunakan model pra-latih sebagai dasar, kemudian dilakukan modifikasi dan pembaruan parameter agar sesuai dengan *dataset* pada masalah baru. Sebagai contoh, model pra-latih yang dirancang untuk mengenali mobil dapat digunakan kembali untuk mengenali truk [7].

2.2. Support Vector Machine (SVM)

Support Vector Machine (SVM) adalah algoritma pembelajaran terarah (*supervised learning*) untuk klasifikasi dan regresi. SVM bekerja dengan memaksimalkan margin dan menggunakan *hyperplane* untuk memisahkan data secara biner. Margin adalah jarak antara *hyperplane* dan titik data terdekat dari setiap kelas, dengan tujuan menemukan *hyperplane* optimal. Parameter seperti *cost* (C) dan jenis *kernel* memengaruhi pembentukan *hyperplane*. *Cost* (C) mengatur penalti untuk kesalahan klasifikasi, sehingga memengaruhi keseimbangan antara margin lebar dan tingkat kesalahan. Nilai C tinggi fokus pada pengurangan kesalahan, tetapi berisiko *overfitting*, sedangkan nilai C rendah memaksimalkan margin, namun dapat menyebabkan *underfitting* [8].



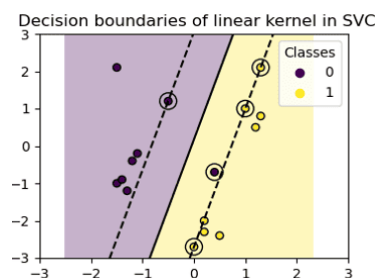
Gambar 1. Hyperplane SVM [9]

a. Kernel Linear

Kernel linear adalah hasil perkalian titik dari sampel *input*:

$$K(x, y) = x \cdot y \quad (1)$$

Kemudian, hal ini diterapkan pada kombinasi dua titik data (sampel) dalam *dataset*. Perkalian titik antara kedua titik tersebut menentukan *cosine similarity* antara keduanya. Semakin tinggi nilainya, semakin mirip kedua titik tersebut [10].



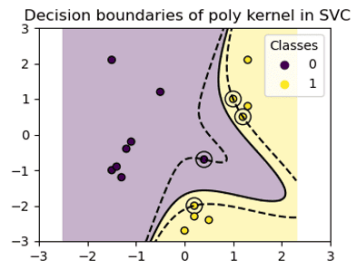
Gambar 2. Kernel Linear [8]

b. Kernel Polynomial

Kernel polynomial mengubah konsep kemiripan antara data. Fungsi *kernel* ini didefinisikan sebagai:

$$K(x_1, x_2) = (\gamma \cdot x_1 x_2 + r)^d \quad (2)$$

Di mana, d adalah *degree*: tingkat *polynomial* yang menentukan kompleksitas *kernel*. γ (*gamma*): mengatur pengaruh setiap sampel pelatihan terhadap batas keputusan. *coef0* adalah istilah bias yang menggeser data ke atas atau ke bawah [10].



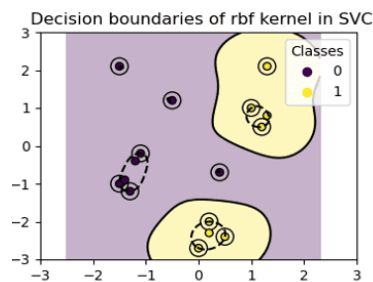
Gambar 3. Kernel Polynomial [8]

c. Kernel RBF (*Radial Basis Function*)

Kernel RBF juga dikenal sebagai *kernel gaussian*, adalah *kernel default* pada *Support Vector Machines* di *scikit-learn*. *Kernel* ini mengukur kemiripan antara dua titik data dalam dimensi tak hingga, dan menentukan klasifikasi berdasarkan mayoritas. Fungsi *kernel* ini didefinisikan sebagai [10]:

$$K(x_1, x_2) = \exp(-\gamma \cdot ||x_1 - x_2||^2) \quad (3)$$

Dimana γ (*gamma*) mengatur pengaruh setiap sampel pelatihan terhadap batas keputusan. Semakin besar jarak *Euclidean* antara dua titik, nilai fungsi *kernel* mendekati nol, menunjukkan bahwa kedua titik kemungkinan besar tidak mirip.



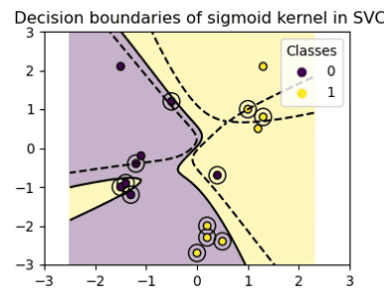
Gambar 4. Kernel RBF [8]

d. Kernel Sigmoid

Fungsi *kernel sigmoid* didefinisikan sebagai:

$$K(x_1, x_2) = \tanh(\gamma \cdot x_1 x_2 + r) \quad (4)$$

Di mana γ (*gamma*) mengontrol pengaruh setiap sampel pelatihan terhadap batas keputusan. *coef0* merupakan istilah bias yang menggeser data ke atas atau ke bawah. *Kernel sigmoid* menghitung kemiripan antara dua titik data menggunakan fungsi *tangens* hiperbolik (*tanh*). Fungsi ini menskalakan dan, jika diperlukan, menggeser hasil *dot product* antara dua titik data [10].



Gambar 5. Kernel Sigmoid [8]

2.2. Preprocessing

Preprocessing data adalah tahap penting dalam pembelajaran mesin. Misalnya, jika gambar objek selalu vertikal, program mungkin kesulitan mengenali objek yang sama jika tampil horizontal. Data yang "bersih" dianggap konsisten dan invarian terhadap translasi, rotasi, serta skala, sehingga program dapat mengenali objek akurat terlepas dari posisi atau orientasinya [11].

a. Noise Removal

Noise removal atau *image denoising* merupakan aspek penting dalam pemrosesan citra, terutama untuk meningkatkan kualitas gambar dalam bidang seperti *medical imaging*, *remote sensing*, dan *digital forensics*. *Noise*, yaitu sinyal yang tidak diinginkan, dapat menurunkan kualitas gambar dan menghambat pengenalan citra yang akurat. *Noise* biasanya muncul karena masalah perangkat keras, operasi perangkat lunak (misalnya konversi format, kompresi), atau faktor lingkungan seperti cacat manufaktur dan *bit errors* [12].

b. Skeletonization

Skeletonization adalah teknik visi komputer untuk menyederhanakan representasi objek sambil menjaga struktur dan sifat geometrisnya. Proses ini menghasilkan "*skeleton*" yang mempermudah analisis dan meningkatkan efisiensi komputasi untuk aplikasi seperti pengenalan karakter, pelacakan gerakan, dan deteksi kabel [13].

c. Region of Interest (ROI)

ROI digunakan untuk membatasi wilayah yang akan digunakan untuk menghemat proses komputasi, penyimpanan dan pengambilan keputusan secara cerdas. Implementasi ROI yang dikembangkan secara manual, menyebabkan berkurangnya tingkat akurasi. Pembuatan ROI dapat dilakukan secara otomatis dengan mendeteksi bagian terluar pada bagian atas dan bawah objek, sehingga dapat meningkatkan akurasi. ROI yang ditempatkan sebelum proses pengolahan citra berguna agar daerah yang dilakukan proses pengolahan citra berfokus pada frame ROI dan bukan pada seluruh frame [14].

d. ImageDataGenerator

ImageDataGenerator berfungsi dengan menghasilkan variasi gambar input melalui berbagai transformasi seperti *rotations*, *flips*, *shifts*, *zooms*, dan *shears*. Transformasi ini dilakukan secara *real-time* selama pelatihan, menciptakan variasi *dataset* yang menyerupai kondisi nyata tanpa memerlukan data berlabel tambahan. Dengan memperluas ukuran *dataset* secara efektif, teknik ini membantu model menjadi lebih mampu melakukan generalisasi dan mengurangi risiko *overfitting* [15].

2.5. Ekstraksi Fitur

Ekstraksi fitur merupakan salah satu tahapan dalam pengolahan citra (*image processing*) yang bertujuan untuk mendapatkan informasi mengenai karakteristik dan ciri-ciri tertentu dari sebuah

citra. Proses ini sangat penting dilakukan agar dapat mengidentifikasi informasi utama yang mampu memberikan gambaran dan interpretasi terkait sebuah objek [16].

a. *Red Green Yellow (RGY)*

Metode ekstraksi fitur RGY (*Red, Green, Yellow*) digunakan untuk menganalisis tanda tangan tumpang tindih guna menentukan keasliannya. Proses ini menghitung tiga fitur dari area tumpang tindih untuk klasifikasi tanda tangan [5]:

- *Red (R)*: Jumlah piksel unik pada tanda tangan uji (Q) yang tidak ada pada tanda tangan referensi, ditandai dengan warna merah.
- *Yellow (Y)*: Jumlah piksel yang sama pada kedua tanda tangan, menunjukkan kemiripan, ditandai dengan warna kuning.
- *Green (G)*: Jumlah piksel unik pada tanda tangan referensi yang tidak ada pada tanda tangan uji, ditandai dengan warna hijau.

b. *Grey Level Co-occurrence Matrix (GLCM)*

Grey Level Co-occurrence Matrix (GLCM) adalah salah satu metode yang digunakan untuk ekstraksi fitur tekstur, khususnya pada penghitungan tekstur tingkat kedua. Pada tingkat pertama, penghitungan tekstur hanya berdasarkan nilai statistik piksel dari gambar asli, seperti varians, tanpa mempertimbangkan hubungan antara piksel yang berdekatan. Sebaliknya, pada tingkat kedua, hubungan antar pasangan dua piksel dalam gambar diperhitungkan [17]. Berikut adalah fitur-fitur dari GLCM [17]:

- *Contrast*, mengacu pada tingkat perbedaan warna keabuan antara piksel dalam suatu citra, yang dapat dihitung menggunakan rumus (5).

$$Contrast = \sum_i^{ng} \sum_j^{ng} (i - j)^2 p(i, j) \quad (5)$$

- *Correlation*, adalah pengukuran hubungan antara satu piksel dengan piksel-piksel tetangganya dalam suatu citra, yang dihitung dengan rumus (6).

$$Correlation = \sum_i^{ng} \sum_j^{ng} (p(i - \mu_i)(j - \mu_j) p(i, j)) / (\sigma_x \sigma_y) \quad (6)$$

- *Energy*, merupakan tingkat keseragaman piksel dalam sebuah citra, yang dihitung dengan menjumlahkan elemen-elemen pada matriks *co-occurrence*, sesuai rumus (7).

$$Energy = \sqrt{ASM} \quad (7)$$

- *Homogeneity* menunjukkan seberapa mirip satu piksel dengan piksel lain di sekitarnya. Nilainya dihitung menggunakan rumus (8).

$$Homogeneity = \sum_i^{ng} \sum_j^{ng} p(i, j) / (1 + (i - j)^2) \quad (8)$$

- *Dissimilarity* adalah ukuran ketidakmiripan tekstur. Nilainya akan lebih kecil jika tekstur lebih seragam, dan sebaliknya. Rumus perhitungannya ditunjukkan pada (9).

$$Disimilarity = \sum_i^{ng} \sum_j^{ng} |i - j| p(i, j) \quad (9)$$

- *Angular Second Moment (ASM)* adalah ukuran keseragaman pada piksel dalam sebuah citra. Nilai ASM akan tinggi jika kemungkinan kemiripan antara piksel juga tinggi. Perhitungannya diberikan oleh rumus (10).

$$ASM = \sum_i^{ng} \sum_j^{ng} (p(i, j))^2 \quad (10)$$

Keterangan: p = probabilitas (0-1) yaitu elemen matriks; (i, j) = pada baris ke- i dan kolom ke- j ; ng = frekuensi tingkat keabuan citra; μ = nilai rata rata (i, j) ; σ = standar deviasi

2.5. Desain Penelitian

Proses yang digambarkan dalam *flowchart* diawali dengan memasukkan data tanda tangan sebagai *input* utama ke dalam sistem. Data ini akan digunakan dalam rangkaian tahapan analisis untuk verifikasi tanda tangan. Setelah data dimasukkan, sistem akan mengevaluasi apakah proses *preprocessing* akan diterapkan pada data tanda tangan tersebut.

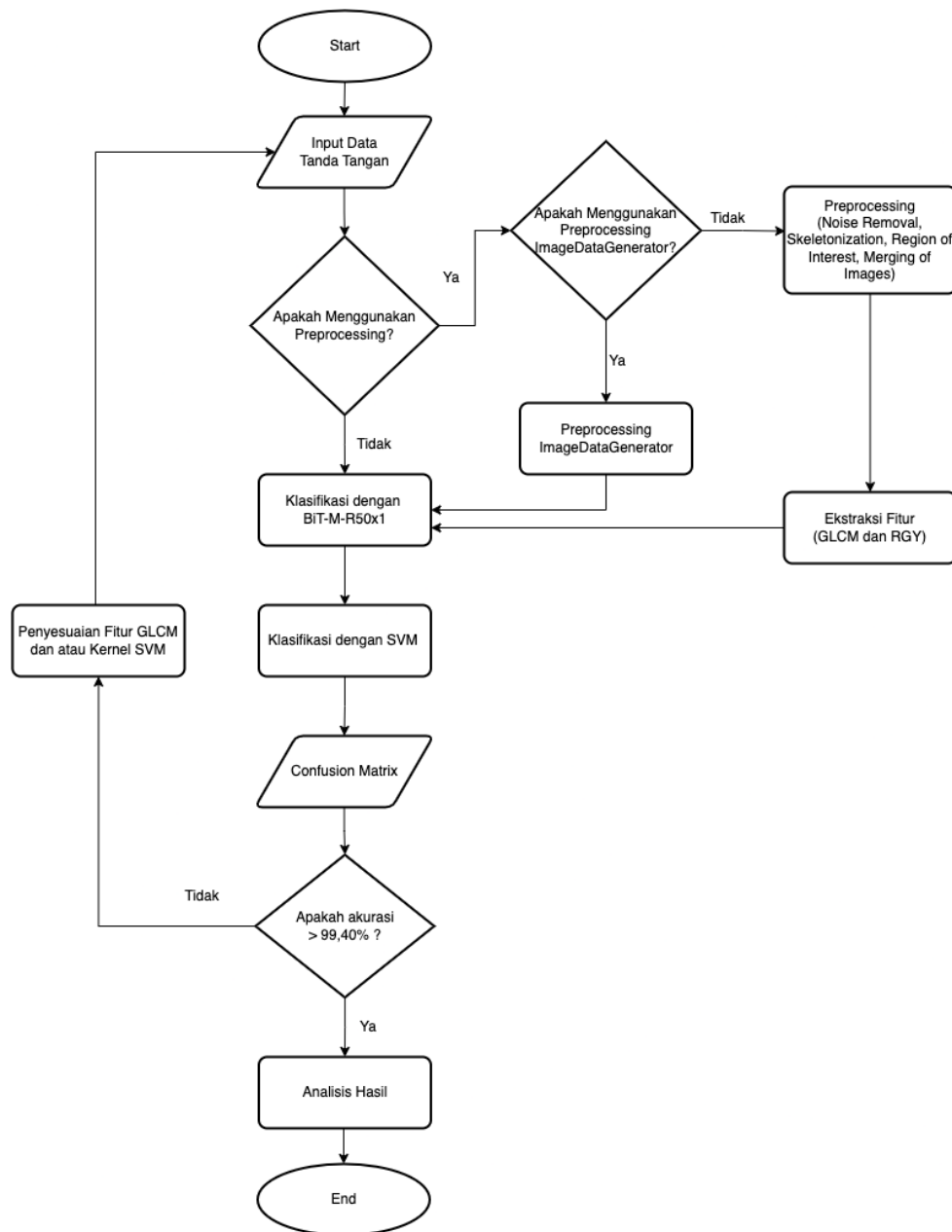
Jika pengguna atau sistem memilih untuk tidak menggunakan *preprocessing*, maka data tanda tangan akan langsung dikirimkan ke tahap klasifikasi utama, yang menggunakan model *BiT-M-R50x1* sebagai dasar pemrosesan. Pada tahap ini, model akan menganalisis karakteristik tanda tangan secara langsung tanpa melalui tahapan pemrosesan tambahan sebelumnya.

Namun, jika *preprocessing* digunakan, sistem akan melanjutkan dengan menentukan metode *preprocessing* yang akan diterapkan pada data tanda tangan tersebut. Pada tahap ini, sistem akan mengecek apakah *preprocessing* akan dilakukan dengan *ImageDataGenerator* atau dengan metode *preprocessing* lainnya.

Jika *preprocessing* yang dipilih adalah *ImageDataGenerator*, sistem akan menerapkan berbagai teknik peningkatan kualitas citra melalui metode ini. *ImageDataGenerator* berfungsi untuk melakukan augmentasi data, dalam penelitian ini yang digunakan adalah *rescale*, sehingga model dapat mengenali variasi tanda tangan dengan lebih baik. Setelah proses augmentasi selesai, data yang telah diproses akan langsung dikirimkan ke tahap klasifikasi dengan menggunakan model *BiT-M-R50x1* untuk analisis lebih lanjut.

Jika sistem memilih untuk tidak menggunakan *ImageDataGenerator*, maka *preprocessing* akan dilakukan dengan serangkaian teknik yang lebih spesifik untuk meningkatkan kualitas citra tanda tangan. Beberapa metode yang digunakan dalam tahap ini mencakup *noise removal*, *skeletonization* *region of interest (ROI)* dan *merging of images*.

Setelah *preprocessing* selesai, tahap berikutnya adalah ekstraksi fitur. Pada tahap ini, sistem akan mengekstrak karakteristik penting dari tanda tangan menggunakan dua metode utama, yaitu GLCM (*Gray Level Co-occurrence Matrix*) dan RGY (*Red-Green-Yellow*). GLCM digunakan untuk menganalisis tekstur citra berdasarkan hubungan antar piksel, sedangkan RGY digunakan untuk mengidentifikasi pola warna yang khas dalam tanda tangan. Hasil dari proses ekstraksi fitur ini kemudian digunakan sebagai input untuk tahap klasifikasi menggunakan model *BiT-M-R50x1*.



Gambar 6. Flowchart

Setelah model *BiT-M-R50x1* selesai melakukan klasifikasi awal, sistem akan melanjutkan proses klasifikasi menggunakan *Support Vector Machine* (SVM) sebagai tahap klasifikasi tambahan. Model SVM ini digunakan untuk mengoptimalkan hasil prediksi dengan memisahkan data berdasarkan *hyperplane* yang paling optimal, sehingga tingkat akurasi dalam membedakan tanda tangan asli dan palsu dapat lebih ditingkatkan.

Setelah semua tahap klasifikasi selesai, sistem akan melakukan evaluasi kinerja model dengan membuat *confusion matrix*. *Confusion matrix* ini digunakan untuk mengukur sejauh mana model dapat membedakan tanda tangan asli dan palsu dengan akurat. Dalam tahap evaluasi ini, sistem akan menghitung beberapa metrik penting seperti *accuracy*, *precision*, *recall*, dan *F1-score* untuk memastikan kualitas hasil klasifikasi.

Langkah selanjutnya adalah verifikasi hasil klasifikasi dengan membandingkan akurasi yang diperoleh dengan nilai 99,40%, yaitu akurasi yang telah dicapai dalam penelitian sebelumnya. Jika akurasi yang diperoleh memenuhi atau melebihi 99,40%, maka sistem akan melanjutkan ke tahap analisis hasil akhir, di mana hasil klasifikasi akan dikaji lebih dalam untuk memahami efektivitas model yang digunakan. Proses ini diakhiri dengan penyajian hasil dalam bentuk laporan atau visualisasi untuk pengguna [4].

Namun, jika akurasi yang diperoleh masih berada di bawah 99,40%, akan dilakukan penggunaan fitur GLCM dan kernel SVM yang berbeda jika menggunakan ekstraksi fitur GLCM atau hanya menggunakan kernel SVM yang berbeda jika tidak menggunakan ekstraksi fitur GLCM. Hal ini bertujuan untuk meningkatkan kualitas model dengan memperbarui data pelatihan atau menyesuaikan parameter dalam proses klasifikasi sehingga hasil yang diperoleh dapat lebih akurat.

2.6. *Confusion Matrix*

Confusion Matrix merupakan tabel yang menunjukkan jumlah data uji yang berhasil diklasifikasikan dengan benar maupun salah, sehingga membantu dalam mengevaluasi akurasi sistem klasifikasi. Dengan memanfaatkan *confusion matrix*, kita dapat menganalisis performa sistem secara mendetail serta mengidentifikasi letak kesalahan klasifikasi yang terjadi [18].

Istilah-istilah penting dalam matriks kebingungan (*confusion matrix*) adalah sebagai berikut [19]:

- True Positive* (TP): Hasil prediksi sesuai dengan kenyataan. Artinya, model memprediksi hasil positif, dan nilai sebenarnya juga positif.
- True Negative* (TN): Hasil prediksi negatif sesuai dengan kenyataan. Artinya, model memprediksi hasil negatif, dan nilai sebenarnya juga negatif.
- False Positive* (FP): Prediksi salah, di mana model memprediksi hasil positif, tetapi nilai sebenarnya negatif. Ini sering disebut sebagai "kesalahan positif palsu."
- False Negative* (FN): Prediksi salah, di mana model memprediksi hasil negatif, tetapi nilai sebenarnya positif. Ini sering disebut sebagai "kesalahan negatif palsu."

Untuk rumus perhitungannya berupa *accuracy*, *precision*, *recall* dan *F1-score*, berikut adalah masing masing dari rumus perhitungannya [20]:

- Accuracy*: *Accuracy* menunjukkan seberapa tepat model dalam mengklasifikasikan data dengan benar, yang dapat dihitung menggunakan rumus (11):

$$Accuracy = (TP + TN) / (TP + FP + TN + FN) \quad (11)$$

- Precision*: *Precision* mengukur tingkat kesesuaian antara data asli dengan hasil prediksi yang dihasilkan oleh model, yang dapat dihitung menggunakan rumus (12).

$$Precision = TP / (TP + FP) \quad (12)$$

- Recall*: *Recall* menunjukkan keberhasilan model dalam mengambil informasi yang relevan, yang dapat dihitung menggunakan rumus (13).

$$Recall = TP / (TP + FN) \quad (13)$$

- F1-score*: *F1-score* menggambarkan rata-rata berbobot dari *precision* dan *recall*, yang dihitung menggunakan rumus (14).

$$F1 - score = 2 / (1/Recall + 1/Precision) \quad (14)$$

3. Hasil dan Pembahasan

Sebelum membahas hasil pengujian dari kombinasi model *BiT-M-R50x1* dan *Support Vector Machine* (SVM), pada bagian ini membahas tahapan *preprocessing* dan ekstraksi fitur yang diterapkan dalam penelitian ini. Teknik *preprocessing* yang digunakan mencakup *noise removal*, *skeletonization*, *region of interest* (ROI), *merging of images*, serta *ImageDataGenerator*. Proses ekstraksi fitur meliputi ekstraksi komponen warna RGY dan fitur GLCM (*contrast*, *dissimilarity*, *homogeneity*, *ASM*, *energy*, dan *correlation*). Metode pengolahan data dengan bahasa pemrograman *python* dengan *library* *scikit-learn* dan *tensorflow*. *Scikit-learn* adalah *library open source* pada *python* yang dibuat berdasarkan *NumPy*, *SciPy*, dan *matplotlib*. *Scikit-learn* dapat digunakan untuk klasifikasi, regresi dan *clustering*. Sedangkan *tensorflow* merupakan *library* pada *python* yang dikembangkan oleh *Google* yang biasanya digunakan untuk *machine learning*.



Gambar 7. Dataset tanda tangan

Gambar 7 (a) merupakan sampel dari dataset tanda tangan asli, dan Gambar 7 (b) merupakan sampel dari dataset tanda tangan palsu.



Gambar 8. Dataset tanda tangan setelah *preprocessing noise removal*

Gambar 8 merupakan hasil dari proses *noise removal*. Proses *noise removal* menjaga kualitas bentuk asli dengan lebih bersih, sehingga objek utama lebih mudah dikenali dan diproses pada tahap analisis berikutnya.



Gambar 9. Dataset tanda tangan setelah *preprocessing skeletonization*

Pada proses *skeletonization* pada Gambar 9 di atas, gambar diubah menjadi garis-garis tipis yang mengikuti kontur asli objek. Garis-garis ini mewakili kerangka atau struktur dasar dari objek tersebut.



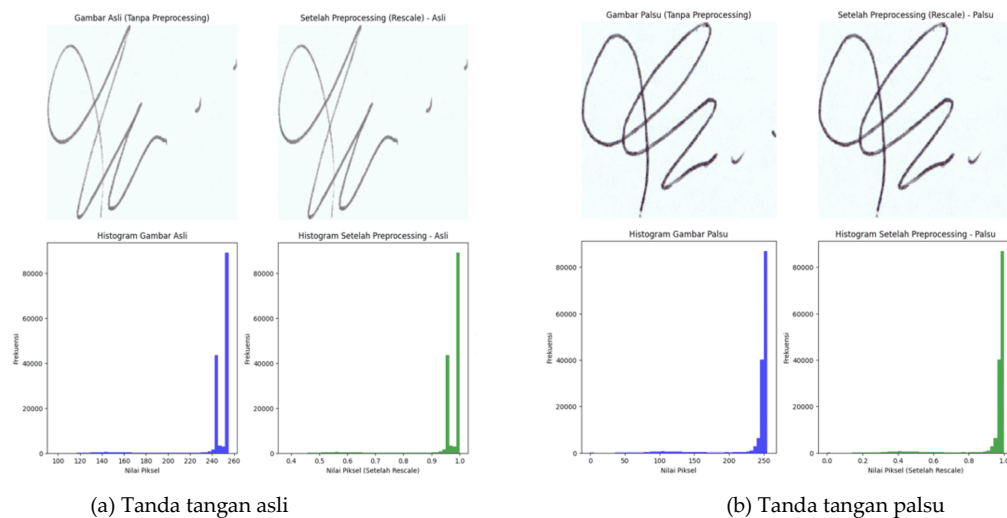
Gambar 10. Dataset tanda tangan setelah *preprocessing region of interest*

Gambar 10 menunjukkan hasil dari *preprocessing region of interest*, gambar akan dipilih mana yang akan dianalisis.



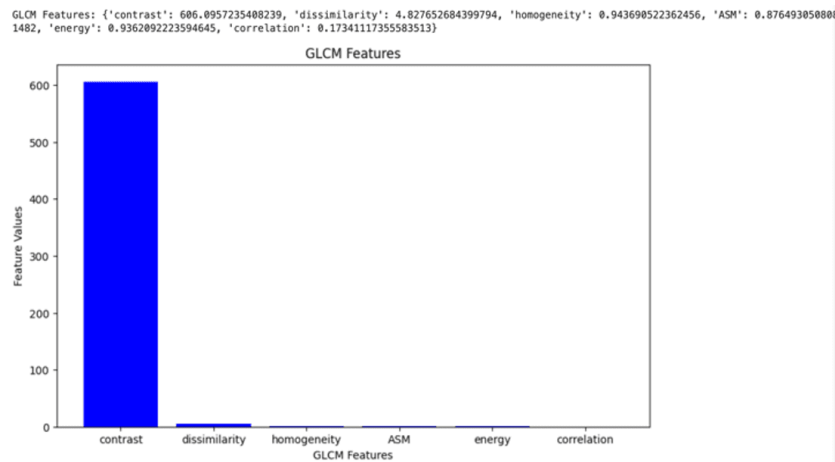
Gambar 11. Dataset tanda tangan setelah *preprocessing merging of images*

Hasil *preprocessing merging of images* pada gambar 11 menunjukkan proses penggabungan antara tanda tangan asli dan tanda tangan palsu dalam format RGY (Red-Green-Yellow). Dalam gambar ini, area yang berwarna merah mewakili piksel yang hanya ada pada tanda tangan asli, sedangkan area berwarna hijau mewakili piksel yang hanya ada pada tanda tangan palsu. Ini memungkinkan kita untuk dengan mudah melihat perbedaan antara kedua tanda tangan tersebut berdasarkan area yang unik pada masing-masing gambar.



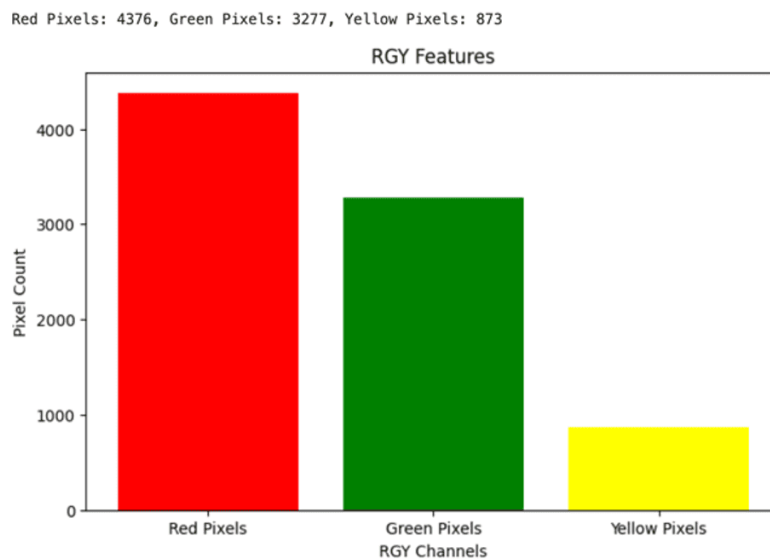
Gambar 12. Dataset tanda tangan setelah *preprocessing ImageDataGenerator*

Pada Gambar 12 menunjukkan proses rescaling pada tanda tangan untuk persiapan dalam model pembelajaran mesin. *Rescaling* ini bertujuan untuk menyesuaikan nilai piksel agar lebih sesuai dengan persyaratan model pembelajaran mesin, yang biasanya bekerja lebih baik dengan nilai yang terstandarisasi atau ter-normalisasi.



Gambar 13. Hasil ekstraksi fitur GLCM

Pada hasil ekstraksi fitur GLCM yang ditunjukkan pada Gambar 13, fitur contrast memiliki nilai yang jauh lebih tinggi dibandingkan fitur-fitur lainnya, yaitu sekitar 606. Nilai contrast yang tinggi ini menunjukkan adanya variasi yang cukup besar antara piksel-piksel yang berdekatan dalam gambar, yang dapat diartikan bahwa tekstur gambar memiliki perubahan intensitas yang tajam. Ini bisa menjadi indikator bahwa tanda tangan memiliki banyak detail halus atau perbedaan intensitas yang signifikan antara garis-garis tanda tangan dan latar belakang.



Gambar 14. Hasil ekstraksi fitur RGY

Hasil ekstraksi fitur RGY pada Gambar 14, menunjukkan jumlah piksel untuk masing-masing kanal warna merah, hijau, dan kuning, dari gambar hasil penggabungan tanda tangan asli dan palsu.

Pada penelitian yang dilakukan oleh Fremmuzar P dan Anna Baita A [21] menyebutkan parameter *cost* (C), *gamma* (γ), dan *degree* (d) memiliki peran penting dalam menentukan kemampuan model dalam menangkap pola pada data. Pada *kernel linear*, mereka menunjukkan bahwa pengaturan parameter C memengaruhi keseimbangan antara margin yang besar dan tingkat kesalahan. Sementara itu, pada *kernel RBF*, pengaturan C dan *gamma* menjadi

kunci dalam menyesuaikan sensitivitas model terhadap data. Untuk *kernel polynomial*, *degree* menjadi faktor tambahan yang memengaruhi kompleksitas model, dan nilai *degree* yang tidak tepat dapat menyebabkan *overfitting*. Sedangkan pada *kernel sigmoid*, penelitian mereka menyoroiti pentingnya pemilihan nilai *C* dan *gamma* secara hati-hati agar model dapat menghasilkan hasil yang optimal tanpa kesalahan signifikan. Pada pengujian penelitian tersebut *kernel linear* mencapai hasil terbaik dengan nilai parameter $C=2$, *kernel RBF* mencapai hasil terbaik dengan nilai parameter $C=4$ dan $\gamma=0.5$, *kernel polynomial* mencapai hasil terbaik dengan nilai parameter $C=2$ dan $d=1$, dan *kernel sigmoid* mencapai hasil terbaik dengan nilai parameter $C=3$ dan $\gamma=2$ [21]. Pada pengujian yang dilakukan penulis, menggunakan parameter *C*, γ dan *d* seperti pada penelitian tersebut.

Pengujian dilakukan dalam dua set data utama, yaitu *validation set* dan *test set*. *Validation set* digunakan untuk mengukur kinerja model selama proses pelatihan, sementara *test set* digunakan untuk mengevaluasi generalisasi model terhadap data baru. Pada setiap skenario, metrik yang dihasilkan dari *confusion matrix* dijadikan acuan untuk membandingkan performa setiap model yang telah diterapkan. Pada penelitian sebelumnya untuk pelatihan menggunakan 80% dari *dataset training* dan validasi menggunakan 20% dari data *training* [4].

Pada Tabel 1 berikut merupakan hasil pengujian menggunakan *preprocessing noise removal*, *skeletonization*, *region of interest* dan *merging of images* ditambah dengan ekstraksi fitur RGY dan GLCM. Hasil pengujian pada Tabel 1 hanya menampilkan tiap *kernel* saja dikarenakan kombinasi *preprocessing* dan ekstraksi fitur tidak memberikan perbedaan kecuali untuk *kernel sigmoid* akan ditampilkan pada Tabel 2 karena terdapat perbedaan antara fitur GLCM yang digunakan. Seperti penelitian yang dilakukan oleh Joseph dan Olugbara (2022), penggunaan beberapa tahap *preprocessing* tidak selalu memberikan peningkatan kinerja, bahkan disebutkan bahwa penambahan beberapa tahap *preprocessing* justru dapat berdampak negatif terhadap performa. Hal ini menunjukkan bahwa dalam beberapa kasus, *preprocessing* dapat menurunkan efektivitas model yang digunakan [22].

Tabel 1. Pengujian 1

No	Kernel SVM	Accuracy	Precision	Recall	F1-Score
1	Linear	0.9860	1.0000	0.9718	0.9857
2	RBF	0.9300	0.8959	0.9718	0.9323
3	Polynomial	0.9860	1.0000	0.9718	0.9857

Pada hasil pengujian Tabel 1 menunjukkan *kernel linear* dan *kernel polynomial* memiliki hasil *accuracy*, *precision*, *recall* dan *F1-Score* yang sama dan *kernel RBF* lebih rendah daripada *kernel linear* dan *kernel polynomial*.

Seperti pada penjelasan sebelumnya pengujian *kernel sigmoid* dengan *preprocessing noise removal*, *skeletonization*, *region of interest* dan *merging of images* ditambah dengan ekstraksi fitur RGY dan GLCM menunjukkan hasil yang berbeda untuk beberapa fitur GLCM yang digunakan.

Tabel 2. Pengujian 2

No	Fitur GLCM	Accuracy	Precision	Recall	F1-Score
1	Contrast	0.8060	0.8326	0.7621	0.7958
2	Dissimilarity	0.8060	0.8326	0.7621	0.7958
3	Homogeneity	0.8100	0.8370	0.7661	0.8000
4	ASM	0.8060	0.8326	0.7621	0.7958
5	Energy	0.8060	0.8356	0.7581	0.7949
6	Correlation	0.8060	0.8356	0.7581	0.7949

Pengujian selanjutnya akan dilakukan pengujian dengan menggunakan preprocessing *ImageDataGenerator* dan tanpa *preprocessing ImageDataGenerator*. Hasil pengujiannya menunjukkan kesamaan antara penggunaan dengan dan tanpa *preprocessing ImageDataGenerator*. Seperti dilihat pada Tabel 2 *kernel linear* memberikan performa terbaik dengan nilai *accuracy*, *precision*, *recall*, dan *F1-score* 100% semua.

Tabel 3. Pengujian 3

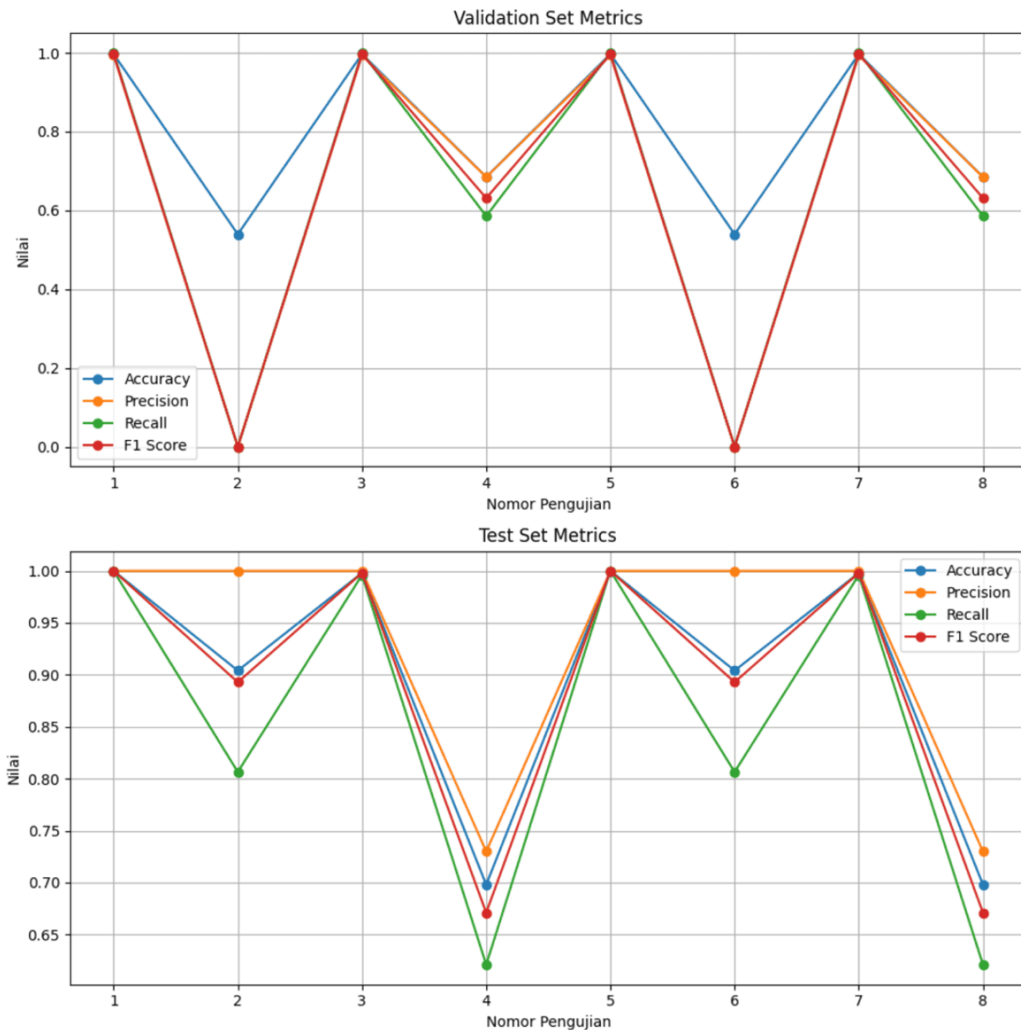
No	Kernel SVM	Accuracy	Precision	Recall	F1-Score
1	Linear	1.0000	1.0000	1.0000	1.0000
2	RBF	0.9040	1.0000	0.8065	0.8929
3	Polynomial	0.9980	1.0000	0.9960	0.9980
4	Sigmoid	0.6980	0.7299	0.6210	0.6710

Pada pengujian selanjutnya akan dilakukan pengujian dengan menggunakan *dataset* yang telah dirusak dengan menambahkan *noise* dan *blur* dengan jumlah *dataset* yang dirusak 25%, 50% dan 75% dari seluruh jumlah *dataset*. Perusakan *dataset* ini bertujuan untuk menguji apakah performa model masih baik ketika *dataset* memiliki *noise* dan *blur*.

Tabel 4. Pengujian 3

No	Tingkat Kerusakan Dataset	Kernel SVM	Accuracy	Precision	Recall	F1-Score
1	25%	Linear	0.9880	0.9879	0.9879	0.9879
2	25%	RBF	0.7420	1.0000	0.4798	0.6485
3	25%	Polynomial	0.9920	0.9919	0.9919	0.9919
4	25%	Sigmoid	0.6720	0.6826	0.6331	0.6569
5	50%	Linear	0.9780	0.9837	0.9718	0.9777
6	50%	RBF	0.5920	1.0000	0.1774	0.3014
7	50%	Polynomial	0.9740	0.9757	0.9718	0.9737
8	50%	Sigmoid	0.6200	0.6343	0.5524	0.5905
9	75%	Linear	0.9780	0.9684	0.9879	0.9780
10	75%	RBF	0.5340	1.0000	0.0605	0.1141
11	75%	Polynomial	0.9760	0.9758	0.9758	0.9758
12	75%	Sigmoid	0.5700	0.5652	0.5766	0.5709

Dari seluruh hasil pengujian *kernel linear* memberikan performa terbaik dibandingkan dengan *kernel* yang lain dan juga meskipun dengan *dataset* telah dirusak dengan diberikan *noise* dan *blur*, *kernel linear* masih lebih baik dibandingkan dengan *kernel* yang lain meskipun terdapat penurunan hasil pengujian dibandingkan dengan *dataset* yang tidak dirusak.



Gambar 15. Grafik hasil pengujian dengan dan tanpa *preprocessing ImageDataGenerator* dan tanpa ekstraksi fitur

4. Kesimpulan

Berdasarkan hasil pengujian yang telah dilakukan, pengujian menggunakan *preprocessing noise removal, skeletonization, region of interest* dan *merging of images* dipadukan dengan ekstraksi fitur RGY dan GLCM menunjukkan hasil yang lebih rendah dan tidak mendapatkan hasil yang diharapkan yaitu melebihi dari hasil penelitian sebelumnya. Pemilihan fitur GLCM yang dilakukan tidak berpengaruh terhadap hasil pengujian terkecuali pada *kernel sigmoid* beberapa fitur GLCM memberikan hasil yang berbeda.

Pemilihan *kernel SVM* terbukti sangat berpengaruh terhadap perbedaan hasil. Hasil terbaik diperoleh ketika model *BiT-M-R50x1* dipadukan dengan SVM menggunakan *kernel linear*, baik dengan maupun tanpa *preprocessing ImageDataGenerator*. Pada pengujian dengan *ImageDataGenerator*, model menunjukkan performa optimal dengan *metrics* pada *validation set* berupa *accuracy* sebesar 0.9970, *precision* 0.9935, *recall* 1.0000, dan *F1 score* 0.9967, sementara pada *test set* semua *metrics* mencapai nilai sempurna yaitu *accuracy, precision, recall*, dan *F1 score* sebesar 1.0000. Hasil serupa juga dicapai pada pengujian tanpa *preprocessing* tambahan, yang kembali menunjukkan *metrics* sempurna pada *test set* serta

performa tinggi pada *validation set* dengan *accuracy* sebesar 0.9970, *precision* 0.9935, *recall* 1.0000, dan *F1 score* 0.9967.

Pengujian model *BiT-M-R50x1* dan SVM tanpa *preprocessing* dan tanpa ekstraksi fitur dengan *dataset* yang telah dirusak yaitu menambahkan *blur* dan *noise* pada *dataset* dengan jumlah kerusakan *dataset* 25%, 50% dan 75% dari seluruh jumlah *dataset* menyebabkan penurunan terhadap hasil pengujian. *Kernel linear* menunjukkan performa terbaik pada semua tingkat kerusakan *dataset*.

Penelitian selanjutnya dapat dilakukan dengan menggunakan dataset yang berbeda serta melakukan tuning terhadap model yang digunakan. Pengujian dengan dataset yang memiliki variasi tanda tangan dan kualitas gambar yang berbeda dari dataset pada penelitian ini dapat membantu mengevaluasi apakah pemilihan model *BiT-M-R50x1* yang dikombinasikan dengan SVM, serta kombinasi *preprocessing*, ekstraksi fitur, dan kernel SVM yang digunakan, tetap menghasilkan kinerja yang baik. Jika terbukti andal pada dataset yang berbeda, model ini berpotensi untuk diterapkan di dunia nyata.

Daftar Pustaka

- [1] Moeljatno, *KUHP: Kitab Undang Undang Hukum Pidana*. Jakarta: Bumi Aksara, 2021.
- [2] H. Kaur and M. Kumar, "Signature identification and verification techniques: state-of-the-art work," *J Ambient Intell Humaniz Comput*, vol. 14, no. 2, pp. 1027–1045, Feb. 2023, doi: 10.1007/s12652-021-03356-w.
- [3] Andrean W. Finaka, "Cara Terhindar Dari Pemalsuan Tanda Tangan." Accessed: Jan. 24, 2024. [Online]. Available: <https://indonesiabaik.id/videografis/cara-terhindar-dari-pemalsuan-tanda-tangan>
- [4] K. Tanvir, "Signature Verification System: Using Big Transfer (BiT-M-R50x1) for Accurate Authentication," *Journal of Image Processing and Intelligent Remote Sensing*, no. 35, pp. 10–19, Sep. 2023, doi: 10.55529/jipirs.35.10.19.
- [5] P. Singh, P. Verma, and N. Singh, "Offline Signature Verification: An Application of GLCM Features in Machine Learning," *Annals of Data Science*, vol. 9, no. 6, pp. 1309–1321, Dec. 2022, doi: 10.1007/s40745-021-00343-y.
- [6] A. B. Singh and K. M. Singh, "Application of error level analysis in image spam classification using deep learning model," *PLoS One*, vol. 18, no. 12 December, Dec. 2023, doi: 10.1371/journal.pone.0291037.
- [7] A. E. Wijaya, W. Swastika, and O. H. Kelana, "Implementasi Transfer Learning Pada Convolutional Neural Network Untuk Diagnosis Covid-19 Dan Pneumonia Pada Citra X-Ray," *SAINSBERTEK Jurnal Ilmiah Sains & Teknologi*, vol. 2, no. 1, Sep. 2021.
- [8] L. N. Farida and S. Bahri, "Klasifikasi Gagal Jantung menggunakan Metode SVM (Support Vector Machine)," *Komputika: Jurnal Sistem Komputer*, vol. 13, no. 2, pp. 149–156, Oct. 2024, doi: 10.34010/komputika.v13i2.11330.
- [9] R. B. Afandi, T. Fajar Nurdiansyah, A. N. Ramadhani, and A. P. Sari, "Implementasi Support Vector Machine Untuk Analisis Sentimen Aplikasi 'MPStore- Super App UMKM,'" *JIP (Jurnal Informatika Polinema)*, vol. 10, no. 4, Aug. 2024.
- [10] Scikit-learn, "Plot classification boundaries with different SVM Kernels." Accessed: Nov. 25, 2024. [Online]. Available: https://scikit-learn.org/1.5/auto_examples/svm/plot_svm_kernels.html
- [11] F. Marpaung, F. Aulia, and R. C. Nabila, *Computer Vision dan Pengolahan Citra Digital*. Surabaya: Pustaka Aksara, 2022.

- [12]Y. Farooq and S. Savaş, "Noise Removal from the Image Using Convolutional Neural Networks-Based Denoising Auto Encoder," *Journal of Emerging Computer Technologies*, vol. 3, no. 1, pp. 21–28, Mar. 2024, doi: 10.57020/ject.1390428.
- [13]T. Wang and Y. Yamakawa, "Edge-Supervised Linear Object Skeletonization for High-Speed Camera," *Sensors*, vol. 23, no. 12, Jun. 2023, doi: 10.3390/s23125721.
- [14]A. H. Pratomo, W. Kaswidjanti, and S. Mu'arifah, "Implementasi Algoritma Region Of Interest (ROI) Untuk Meningkatkan Performa Algoritma Deteksi Dan Klasifikasi Kendaraan," *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIK)*, vol. 7, no. 1, pp. 155–162, 2020, doi: 10.25126/jtiik.202071718.
- [15]S. E. Sorour, A. A. Wafa, A. A. Abohany, and R. M. Hussien, "A Deep Learning System for Detecting Cardiomegaly Disease Based on CXR Image," *International Journal of Intelligent Systems*, vol. 2024, 2024, doi: 10.1155/2024/8997093.
- [16]S. Ayu Rosiva Srg, M. Fahriyal Aldi, M. Ramadhan, and N. Legahati Siregar, "Ekstraksi Fitur Citra Berdasarkan Tekstur Dengan GLCM (Gray Level Co-Occurrence)," *JUTISAL (Jurnal Teknik Informatika Komputer Universal)*, vol. 3, no. 1, Jun. 2023.
- [17]M. A. Lutfia, F. X. A. Setyawan, S. Alam, T. Yulianti, and H. Fitriawan, "Implementasi Ekstraksi Fitur Menggunakan Gray Level Co-Occurrence Matrices (GLCM) dan K-Nearest Neighbor (K-NN) Untuk Klasifikasi Jenis Kain Dasar," in *Seminar Nasional Teknik Elektro (SNTE)*, 2023.
- [18]R. Nurhidayat and K. E. Dewi, "Penerapan Algoritma K-Nearest Neighbor Dan Fitur Ekstraksi N-Gram Dalam Analisis Sentimen Berbasis Aspek," *Jurnal Ilmiah Komputer dan Informatika*, vol. 12, no. 1, Apr. 2023.
- [19]I. Akil and I. Chaidir, "Classification of Heart Disease Diagnoses Using Gaussian Naïve Bayes," *Jurnal Ilmiah Ilmu Komputer Dan Matematika*, vol. 21, pp. 31–36, 2024, doi: 10.33751/komputasi.v21i2.5260.
- [20]A. Sanmorino and D. Palupi Rini, "Building Research Productivity Framework in Higher Education Institution," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 12, no. 6, p. 2021, [Online]. Available: www.ijacsa.thesai.org
- [21]P. Fremmuzar and A. Baita, "Uji Kernel SVM dalam Analisis Sentimen Terhadap Layanan Telkomsel di Media Sosial Twitter," *Komputika: Jurnal Sistem Komputer*, vol. 12, no. 2, pp. 57–66, Sep. 2023, doi: 10.34010/komputika.v12i2.9460.
- [22]S. Joseph and O. O. Olugbara, "Preprocessing Effects on Performance of Skin Lesion Saliency Segmentation," *Diagnostics*, vol. 12, no. 2, Feb. 2022, doi: 10.3390/diagnostics12020344.

This Page Intentionally Left Blank