IMPLEMENTASI METODE TRANSFER LEARNING DAN ALGORITMA RANDOM FOREST PADA IDENTIFIKASI IRIS MATA

SKRIPSI

OLEH:
AKHMAD ROZIQIN
NIM 19610087

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2023

IMPLEMENTASI METODE TRANSFER LEARNING DAN ALGORITMA RANDOM FOREST PADA IDENTIFIKASI IRIS MATA

SKRIPSI

Diajukan Kepada
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang
untuk Memenuhi Salah Satu Persyaratan
dalam Memperoleh Gelar Sarjana Matematika (S.Mat)

Oleh
Akhmad Roziqin
NIM. 19610087

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2023

IMPLEMENTASI METODE TRANSFER LEARNING DAN ALGORITMA RANDOM FOREST PADA IDENTIFIKASI IRIS MATA

SKRIPSI

Oleh
Akhmad Roziqin
NIM. 19610087

Telah Disetujui Untuk Diuji
Malang, 15 Desember 2023

Dosen Pembimbing I	Dosen Pembimbing II
Hisyam Fahmi, M.Kom.	Juhari, M.Si.
NIP. 19890727 201903 1 018	NIP. 19840209 202321 1 010

Mengetahui,
Ketua Program Studi Matematika

Dr. Elly Susanti, M.Sc.
NIP. 19741129 200012 2 005

IMPLEMENTASI METODE TRANSFER LEARNING DAN ALGORITMA RANDOM FOREST PADA IDENTIFIKASI IRIS MATA

SKRIPSI

Oleh
Akhmad Roziqin
NIM. 19610087

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima sebagai Salah Satu Persyaratan
untuk Memperoleh Gelar Sarjana Matematika (S.Mat)

Tanggal 27 Desember 2023

Ketua Penguji	: Prof. Dr. H. Turmudi, M.Si, Ph.D.	.....
Anggota Penguji 1	: Muhammad Khudzaifah, M.Si.	.....
Anggota Penguji 2	: Hisyam Fahmi, M.Kom.	.....
Anggota Penguji 3	: Juhari, M.Si.	.....

Mengetahui,
Ketua Program Studi Matematika

Dr. Elly Susanti, M.Sc
NIP. 19741129 200012 2 005

HALAMAN PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama : Akhmad Roziqin

NIM : 19610087

Program Studi : Matematika

Fakultas : Sains dan Teknologi

Judul Skripsi : Implementasi Metode *Transfer Learning* dan Algoritma *Random Forest* Pada Identifikasi Iris Mata

Menyatakan dengan sebenar-benarnya bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya sendiri, bukan merupakan pengambilan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan dan pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka. Apabila kemudian hari terbukti atau dapat dibuktikan skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 27 Desember 2023

Yang membuat pernyataan,

Akhmad Roziqin

NIM. 19610087

HALAMAN MOTO

"Always Remember Allah SWT,
Remember Why You Started and Finish What You Start"

HALAMAN PERSEMBAHAN

Bismillahirrahmanirrahim, dengan mengucap syukur kepada Allah SWT skripsi ini peneliti persembahkan kepada ibunda Winnuryati, ayahanda Heru Suprayogi, adik Muhammad Khilmi, serta seluruh teman-teman Soulmath 19 dan IMAPAS yang senantiasa mendoakan dan memberi motivasi, nasihat, dan dukungan baik secara materiil maupun moril sehingga peneliti dapat menyelesaikan skripsi ini.

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh

Alhamdulillah, segala puji dan syukur senantiasa peneliti panjatkan kepada Allah *subhanahu wa ta'ala* atas segala rahmat, taufik, dan hidayah yang telah diberikan sehingga penulis dapat menyelesaikan skripsi dengan judul "Implementasi Metode *Transfer Learning* dan *Random Forest* pada Identifikasi Iris Mata". Shalawat dan salam senantiasa tercurahkan kepada Nabi Muhammad *shallallahu 'alaihi wa sallam* yang telah membimbing manusia dalam jalan kebenaran, yakni agama Islam dan dinantikan syafaat beliau kelak di akhirat.

Dalam proses penyelesaian proposal skripsi ini, peneliti banyak mendapat bimbingan, dukungan, dan arahan serta sumbangsih dari berbagai pihak. Oleh karena itu peneliti mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Prof. Dr. M. Zainuddin, M.A., selaku Rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Prof. Dr. Hj. Sri Harini, M.Si., selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Dr. Elly Susanti, M.Sc., selaku Ketua Program Studi Matematika Universitas Islam Negeri Maulana Malik Ibrahim Malang.
4. Hisyam Fahmi, M.Kom., selaku dosen pembimbing I yang telah memberikan berbagai pengetahuan, pengalaman, arahan, nasihat, serta motivasi kepada peneliti.
5. Juhari, M.Si., selaku dosen pembimbing II yang telah memberikan ilmu, nasihat, bimbingan, pengalaman, serta motivasi kepada peneliti.
6. Seluruh dosen Program Studi Matematika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
7. Kedua orang tua serta seluruh keluarga yang senantiasa mendoakan, memberikan semangat, dukungan, nasihat, serta kasih sayang sehingga peneliti dapat menyelesaikan tugas akhir.
8. Rekan peneliti yang berjuang bersama dalam mengerjakan skripsi yang senantiasa memberikan dukungan dan semangat kepada peneliti.

9. Seluruh teman-teman Program Studi Matematika angkatan 2019 dan Ikatan Mahasiswa Pasuruan yang senantiasa memberikan bantuan, dukungan dan semangat kepada peneliti dalam berbagai kondisi.

Semoga Allah *subhanahu wa ta'ala* selalu memberikan balasan atas segala bantuan dan kebaikan yang telah diberikan kepada penulis. Penulis berharap agar laporan ini dapat bermanfaat bagi penulis serta pembaca untuk menambah wawasan keilmuan yang selalu berkembang.

Wassalamu'alaikum Warahmatullahi Wabarakatuh

Malang, 27 Desember 2023

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

ABSTRAK

Roziqin, Akhmad. 2023. **Implementasi Metode *Transfer Learning* dan Algoritma *Random Forest* pada Identifikasi Iris Mata.** Skripsi. Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Hisyam Fahmi, M.Kom. (II) Juhari, M.Si.

Kata Kunci: *Machine Learning*, *Transfer Learning*, *Random Forest*, Klasifikasi, Iris Mata

Perkembangan teknologi saat ini memiliki potensi untuk meningkatkan *cybercrime*. Oleh karena itu, diperlukan sistem keamanan yang memiliki tingkat keamanan tinggi. Salah satu sistem keamanan tertinggi saat ini yaitu biometrik. *Machine learning* merupakan salah satu teknologi yang dapat digunakan untuk menciptakan sistem keamanan biometrik. Iris mata merupakan salah satu media biometrik yang memiliki tingkat keamanan yang tinggi. Salah satu keunggulan iris mata ialah pola yang cenderung tidak berubah seiring bertambahnya usia. Dalam pengembangan sistem keamanan biometrik, pada penelitian ini digunakan metode *transfer learning* dan algoritma *random forest*. *Transfer learning* merupakan metode *machine learning* yang dilakukan dengan menggunakan model yang telah dilatih sebelumnya. Pada penelitian ini, model *transfer learning* yang digunakan adalah model AlexNet. *Random forest* merupakan metode *machine learning* yang dilakukan dengan menggunakan lebih dari satu *decision tree*. Hasil klasifikasi *random forest* diperoleh dengan menghitung rata-rata hasil klasifikasi dari setiap *decision tree*. Data yang digunakan dalam penelitian ini ialah data berupa citra iris mata yang diambil dari dataset CASIA V1 dengan jumlah data sebanyak 756 citra yang terbagi dalam 108 kelas. Proses evaluasi dilakukan menggunakan algoritma K-*fold Cross Validation* dengan nilai $k = 6$. Pada penelitian ini, diperoleh hasil akurasi akhir sebesar 85,98%.

ABSTRACT

Roziqin, Akhmad. 2023. **The Implementation of Transfer Learning Method and Random Forest Algorithm on Iris Identification.** Undergraduate Thesis. Mathematics Department, Fakulty of Science and Technology, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Advisors: (I) Hisyam Fahmi, M.Kom. (II) Juhari, M.Si.

Keywords: Machine Learning, Transfer Learning, Random Forest, Classification, Iris

Current technological developments have the potential to increase cybercrime. Therefore, a security system that has a high level of security is needed. One of the highest security systems today is biometric. Machine learning is one of the technologies that can be used to create a biometric security system. Iris is one of the biometric media that has a high level of security. One of the advantages of the iris is that the pattern tends not to change with age. In developing a biometric security system, this research used transfer learning method and random forest algorithm. Transfer learning is a machine learning method that is performed using a pre-trained model. In this research, the transfer learning model used is the AlexNet model. Random forest is a machine learning method performed using more than one decision tree. Random forest classification results are obtained by calculating the average classification results of each decision tree. The data used in this study is data in the form of iris images taken from the CASIA V1 dataset with a total of 756 images divided into 108 classes. The evaluation process is carried out using the K-fold Cross Validation algorithm with a value of $k = 6$. In this study, the final accuracy result was 85.98%.

مستخلص البحث

رازقين، احمد. ٢٠٢٣. **تطبيق طريقة التعلم النقلي وخوارزمية الغابة العشوائية في التعرف على قزحية العين.** البحث الجامعي. قسم الرياضيات،كلية العلوم والتكلنولوجيا، جامعة مولانا مالك إبراهيم الإسلامية الحكومية مالانج. المشرف: (١) هشام فهمي، الماجستير، (٢) جوهري، الماجستير.

الكلمات المفتاحية: التعليم الآلي، نقل التعلم، غابة عشوائية، تصنيف، قزحية العين

ويمكن للتطورات التكنولوجية الحالية أن تزيد من الجريمة السيبرانية. لذلك، هناك حاجة إلى نظام أمني يتمتع بمستوى عالٍ من الأمن. يمكن تنفيذ أحد أعلى أنظمة الأمان اليوم باستخدام القياسات هو الحيوية. التعلم الآلي هو أحد التقنيات التي يمكن استخدامها لإنشاء نظام أمان بيومتري. إيريس هي واحدة من وسائط المفياس الحيوي التي تتمتع بمستوى عالٍ من الأمان. إحدى مزايا القزحية هي أن النمط لا يتغير مع تقدم العمر. عند تطوير نظام أمان بيومتري، استخدم هذا البحث طريقة التعلم النقلي وخوارزمية الغابات العشوائية. التعلم النقلي هو طريقة تعلم العالي يتم تنفيذها باستخدام نموذج مدرب مسبقًا. في هذا البحث، نموذج التعلم النقلي المستخدم هو نموذج ***AlexNet***. الغابة العشوائية هي طريقة تعلم آلي يتم تنفيذها باستخدام أكثر من شجرة قرار واحدة. يتم الحصول على نتائج التصنيف العشوائي للغابات عن طريق حساب متوسط نتائج التصنيف لكل شجرة قرار. البيانات المستخدمة في هذ البحث هي بيانات على شكل صور قزحية العين مأخوذة من مجموعة بيانات ***CASIA V1*** بإجمالي ٧٥٦ صورة مقسمة إلى ١٠٨ فئات. تتم عملية التقييم باستخدام خوارزمية ***K-fold Cross Validation*** بقيمة ٦ k =. في هذ البحث، كانت نتيجة الدقة النهائية ٨٥,٩٨٪.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Teknologi *machine learning* menyediakan banyak manfaat dalam berbagai bidang. Salah satu penerapan *machine learning* digunakan dalam proses klasifikasi atau identifikasi objek berdasarkan karakteristik numerik objek tersebut. *Machine learning* merupakan salah satu bagian dari *artificial intelligence* (kecerdasan buatan) yang dapat digunakan untuk melatih suatu sistem atau model untuk mempelajari data tanpa menuliskan algoritma secara eksplisit (Kusuma, 2020). Seiring perkembangannya, *machine learning* memiliki berbagai metode atau algoritma yang dapat digunakan dalam proses identifikasi. Salah satu metode yang telah dikembangkan dalam *machine learning* adalah *transfer learning* dan *Random Forest*.

Dalam bidang *machine learning*, *transfer learning* didefinisikan sebagai metode ekstraksi fitur berbasis *convolutional neural network* yang telah dikembangkan dan dilatih sebelumnya dengan data besar. Keunggulan *transfer learning* yaitu memiliki performa yang tetap baik walau diterapkan ke dalam dataset yang cenderung lebih sedikit (Jauhari, 2020). Penggunaaan *transfer learning* dapat mengurangi biaya komputasi yang perlu dilakukan. Pada penelitian ini, penulis menggunakan *transfer learning* untuk ekstraksi fitur citra. Penerapan metode *transfer learning* dalam proses ekstraksi fitur diharapkan dapat menghasilkan tingkat akurasi yang baik.

Random forest pada dasarnya merupakan pengembangan dari metode *decision tree*, yaitu metode *machine learning* yang digunakan untuk mengambil keputusan berdasarkan pohon keputusan. *random forest* menggunakan lebih dari satu *decision tree* sebagai model pengambilan keputusan berdasarkan hasil *voting* terbesar dari masing-masing *decision tree* (Breiman, 2001). Keunggulan penggunaan metode *random forest* yaitu memiliki tingkat *error* yang cenderung rendah.

Pada penelitian ini, metode *transfer learning* akan diimplementasikan bersama dengan metode *random forest* sebagai teknologi pendukung keamanan data. Penggabungan kedua metode ini dilakukan dengan tujuan untuk menghasilkan model *machine learning* dengan tingkat akurasi yang tinggi sehingga dapat menghasilkan teknologi keamanan yang baik. Salah satu teknologi yang umum digunakan dalam keamanan data adalah teknologi biometrik. Biometrik merupakan teknologi yang digunakan untuk mengenali karakteristik individu baik melalui sidik jari, wajah, suara, atau iris (Sumijan, dkk., 2021). Penggunaan teknologi biometrik untuk keamanan data dinilai efektif dikarenakan tiap individu memiliki karakteristik sidik jari, wajah, suara, dan iris yang berbeda-beda.

Pada dasarnya, manusia memiliki karakteristik yang berbeda-beda. Hal itu dapat dilihat dari DNA, sidik jari, wajah, suara, maupun iris mata manusia yang berbeda-beda. Hal ini sesuai dengan firman Allah SWT dalam QS. At-Tin ayat 4 yang artinya (Kemenag RI, 2022):

"Sungguh, Kami telah menciptakan manusia dalam bentuk yang sebaik-baiknya" (QS. At-Tin:4)

Berdasarkan ayat tersebut, maka dapat diketahui bahwa setiap manusia diciptakan dalam bentuk yang sempurna dan sebaik-baiknya. Setiap bagian dari diri

manusia memiliki ciri-ciri dan karakteristik yang berbeda-beda sebagai salah satu bentuk kekuasaan Allah SWT.

Iris mata merupakan salah satu karakteristik individu yang dapat digunakan untuk keperluan biometrik dalam pengamanan data. Iris mata merupakan organ tubuh yang bersifat internal dan terlindungi oleh kornea (Patel, 2008). Oleh karena itu, iris mata merupakan organ tubuh yang sulit untuk ditiru sehingga memiliki tingkat keamanan yang sangat tinggi. Selain itu, iris mata memiliki tingkat perubahan yang cenderung sedikit tiap tahunnya dibandingkan dengan wajah, sidik jari, maupun suara (Hakim, 2022). Oleh karena itu, pada penelitian ini penulis memilih menggunakan iris mata sebagai objek penelitian.

Pada penelitian sebelumnya, proses pengenalan iris mata telah dikaji dengan menggunakan metode *transfer learning* dengan model AlexNet (G. Alaslani & A. Elrefaei 2018) serta klasifikasi menggunakan algoritma *support vector machine*. Keunggulan model AlexNet yaitu memiliki tingkat akurasi yang lebih baik dibanding model *transfer learning* lainnya. AlexNet sudah dilatih menggunakan *dataset* ImageNet dengan total 1,2 juta citra yang terbagi menjadi 1000 kelas. Penelitian tersebut mendapatkan hasil akurasi sebesar 85% dengan waktu 0.02 detik menggunakan dataset CASIA-Iris-V1 setelah normalisasi.

Pada penelitian ini akan digunakan algoritma *random forest* menggantikan algoritma *support vector machine* pada penelitian sebelumnya dalam proses klasifikasi citra iris mata. Dengan kombinasi ekstraksi fitur menggunakan *transfer learning* model AlexNet beserta klasifikasi menggunakan algoritma *random forest*, diharapkan penelitian ini dapat menghasilkan tingkat akurasi yang tinggi dalam identifikasi iris mata.

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang di atas maka diambil rumusan masalah sebagai berikut.

1. Bagaimana proses implementasi metode *transfer learning* dan *random forest* pada identifikasi iris mata?
2. Bagaimana tingkat akurasi yang dihasilkan dari implementasi metode *transfer learning* dan *random forest* pada identifikasi iris mata?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah di atas maka tujuan penelitian ini adalah sebagai berikut.

1. Mengetahui proses implementasi metode *transfer learning* dan *random forest* pada identifikasi iris mata.
2. Mengetahui tingkat akurasi yang dihasilkan dari implementasi metode *transfer learning* dan *random forest* pada identifikasi iris mata.

1.4 Manfaat Penelitian

Manfaat dari penelitian ini berdasarkan uraian-uraian di atas dipaparkan sebagai berikut:

1. Bagi Peneliti

   Menambah wawasan dan ilmu pengetahuan mengenai implementasi metode *transfer learning* dan *random forest* pada identifikasi iris mata beserta tingkat akurasinya.

2. Bagi Pembaca

Memberikan wawasan dan ilmu pengetahuan terkait implementasi metode *transfer learning* dan *random forest* pada identifikasi iris mata.

1.5 Batasan Masalah

Batasan masalah yang digunakan pada penelitian ini adalah sebagai berikut:

1. File citra iris mata memiliki ekstensi *bmp*.
2. Perancangan program menggunakan bahasa pemrograman Python.
3. *Transfer learning* yang digunakan menggunakan model AlexNet

1.6 Definisi Istilah

Terdapat beberapa istilah yang digunakan pada penelitian ini sebagai berikut:

Data Latih	: Data yang digunakan untuk melatih algoritma program untuk mendapatkan model yang sesuai
Data Uji	: Data yang digunakan untuk menguji model dan mengetahui tingkat keakuratan
Learning Rate	: Salah satu parameter *training* untuk menghitung nilai koreksi bobot pada proses pelatihan
Preprocessing	: Proses pengolahan data untuk mengubah data mentah ke dalam bentuk yang lebih mudah dipahami sistem
Transfer Learning	: Metode ekstraksi fitur berbasis *convolutional neural network* yang telah dikembangkan dan dilatih sebelumnya terhadap data besar

Citra	:	Citra merupakan suatu gambaran yang merepresentasikan suatu objek
Iris	:	Organ bagian dari mata yang berada di antara pupil dan sklera (bagian putih pada mata)

BAB II

KAJIAN PUSTAKA

2.1 Teori Pendukung

2.1.1 *Machine Learning*

Machine learning merupakan salah satu bagian dari *artificial intelligence* (kecerdasan buatan) yang dapat digunakan untuk melatih suatu sistem atau model untuk mempelajari data tanpa menuliskan algoritma secara eksplisit (Kusuma, 2020). Dalam penerapannya *machine learning* sering digunakan dalam proses klasifikasi, yaitu pengelompokkan data berdasarkan kategori tertentu untuk selanjutnya digunakan untuk mendeteksi kategori dari data baru.

Secara umum, *machine learning* dibagi menjadi tiga jenis pendekatan yaitu *supervised learning*, *unsupervised learning*, dan *reinforcement learning* (Murphy, 2012).

1. *Supervised Learning*

   Supervised learning merupakan jenis *machine learning* yang dilakukan menggunakan data yang telah diberi label. Tujuan dari pendekatan ini adalah untuk mempelajari pola hubungan antara *input* x dengan *output* y dimana $y \in \{1, \ldots, C\}$ dengan C adalah banyak kelas atau kategori. Pada *supervised learning*, *output* y juga umum disebut sebagai label dari *input* x. *Supervised learning* pada umumnya sering digunakan untuk keperluan klasifikasi, pengenalan pola, atau regresi.

   Pada *supervised learning*, diperlukan dua data yaitu data latih dan data uji. Data latih merupakan pasangan *input-output* $D = \{(x_i, y_i)\}_{i=1}^{N}$ dimana

merupakan banyak data latih. Data latih merupakan data yang digunakan untuk melatih sistem untuk mendapatkan model terbaik. Hasil dari proses latih ini selanjutnya dapat digunakan untuk memprediksi *output* baik berupa kelas atau kategori dari data baru. Data baru yang digunakan untuk menguji model *supervised learning* disebut sebagai data uji.

Setiap data x_i yang digunakan dalam *supervised learning* merupakan sebuah vektor dengan dimensi D yang merepresentasikan suatu ciri dari data tersebut. Vektor ini juga sering disebut sebagai fitur. Pada umumnya, data x_i yang digunakan berasal dari objek terstruktur seperti citra, teks, grafik, dll.

2. *Unsupervised Learning*

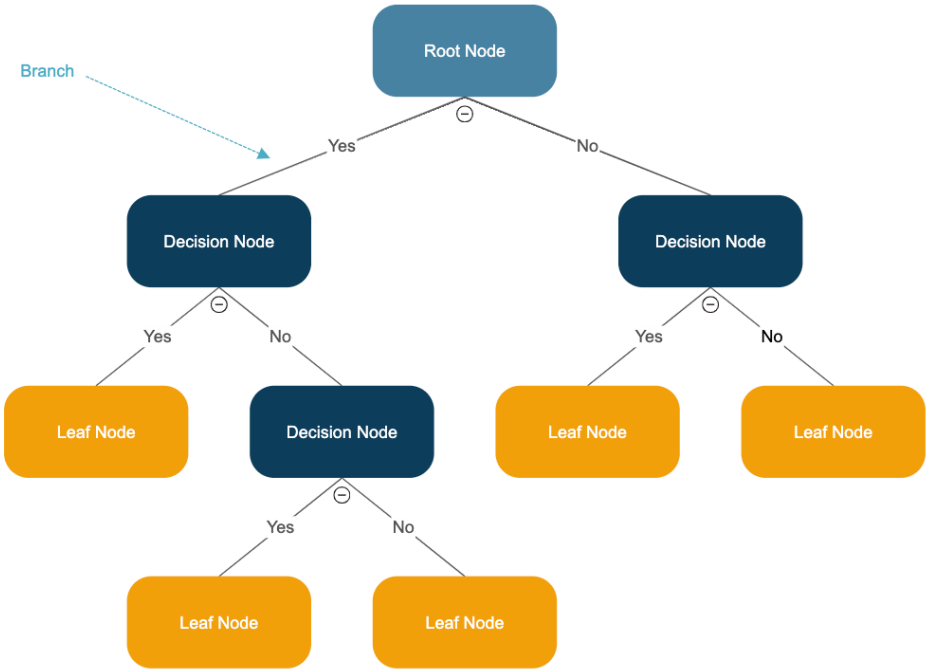
Unsupervised learning merupakan jenis *machine learning* yang dilakukan terhadap data yang tidak memiliki label. Tujuan dari *unsupervised learning* adalah untuk menemukan pola dari sebuah data. *Unsupervised learning* sering digunakan untuk mengelompokkan data berdasarkan kemiripan atau sering dikenal sebagai *clustering*. Berbeda dengan *supervised learning*, *unsupervised learning* hanya memerlukan satu data yaitu $D = \{x_i\}_{i=1}^{N}$ dengan N merupakan banyak data.

3. *Reinforcement Learning*

Pada dasarnya, *reinforcement learning* merupakan jenis *machine learning* yang mempelajari bagaimana cara bertindak atau berperilaku terhadap sesuatu. Berbeda dengan dua jenis *machine learning* sebelumnya, *reinforcement learning* memerlukan sinyal yang dapat bernilai positif atau negatif untuk mengambil tindakan atau perilaku.

2.1.1.1 *Random Forest*

Random forest merupakan suatu algoritma *machine learning* yang populer digunakan dalam klasifikasi data besar. Pada dasarnya, *random forest* merupakan pengembangan dari algoritma *decision tree* (pohon keputusan).



Gambar 2.1 Struktur *Decision Tree*
Sumber: SmartDraw

Decision tree merupakan algoritma yang memiliki struktur seperti pohon yang digunakan untuk mengambil sebuah keputusan (Arvianna, 2022). Pada *decision tree* terdapat banyak simpul yang disebut sebagai *node*. *Decision tree* memiliki tiga komponen utama yaitu *root node* (akar), *branches* (ranting), dan *leaf node* (daun) dapat dilihat pada Gambar 2.1. *Root node* merupakan sebuah simpul yang menjadi akar dari sebuah *decision tree*. Dengan demikian, *root node* merupakan pengondisian pertama dalam pengambilan keputusan menggunakan *decision tree*. *Branches* merupakan penghubung antar *node* sebelumnya ke *node-node* selanjutnya. Jika pada sebuah data memenuhi kondisi yang benar/salah dalam

suatu *node*, selanjutnya data tersebut akan dicek lebih lanjut kepada *node* selanjutnya yang dihubungkan oleh *branches* yang bernilai *yes*/*no*. *Leaf node* merupakan *node* terakhir dalam *decision tree*. Keputusan terakhir atau hasil klasifikasi sebuah data menggunakan *decision tree* dapat diperoleh dengan melihat *leaf node* yang dipenuhi oleh data tersebut.

Secara matematis, struktur *decision tree* dibuat dengan menghitung nilai *entropy* dan nilai *information gain* (Hakim, 2019). Nilai *entropy* merepresentasikan tingkat ketidakpastian terhadap variabel acak. Nilai *entropy* dapat dihitung menggunakan Persamaan (2.1).

$$\text{Entropy}(S) = \sum_{i=1}^{c} -p_i \log_2 p_i \tag{2.1}$$

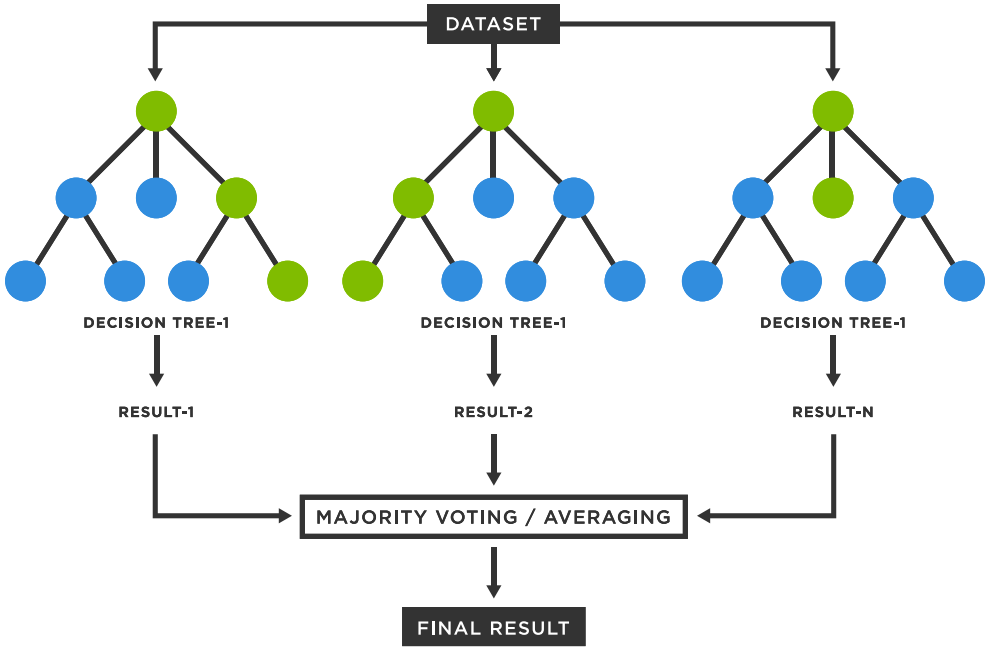
dimana S merepresentasikan atribut dari suatu fitur, p_i merupakan nilai peluang data dari suatu kelas dengan atribut S dan c merupakan banyak kelas. Setelah dihitung nilai *entropy* dari setiap atribut dalam suatu fitur, maka selanjutnya dapat dihitung nilai *information gain* dari fitur tersebut. Nilai ini digunakan untuk menentukan urutan/struktur *decision tree*. Adapun nilai *information gain* dapat dihitung menggunakan Persamaan (2.2).

$$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} \text{Entropy}\,(S_v) \tag{2.2}$$

dimana S merepresentasikan kelas dari data, A adalah fitur, $Entropy(S)$ merupakan nilai *entropy* dari kelas S, $Values(A)$ merupakan banyak atribut yang terdapat dalam A, dan S_v merepresentasikan setiap atribut dari A. Setelah dihitung nilai *information gain* dari masing-masing fitur, selanjutnya dapat ditentukan struktur *decision tree* berdasarkan nilai *information gain* tersebut. Fitur dengan nilai *information gain* terbesar akan menjadi *root node* dari *decision tree*. Selanjutnya

untuk memilih *branches* dari *root node* dapat dilakukan dengan mengulangi langkah yang sama sehingga mendapatkan fitur dengan nilai *information gain* terbesar dan seterusnya.

Kelemahan penggunaan *decision tree* sebagai algoritma klasifikasi adalah rentan terjadinya *overfitting*, yaitu ketidaksesuaian hasil prediksi terhadap data baru. Hal ini terjadi dikarenakan model *decision tree* dibuat dengan berdasarkan data latih yang ada. Sehingga model yang dihasilkan cenderung hanya dapat memprediksi data yang digunakan untuk latihan saja. Untuk mengatasi kelemahan ini, maka dikembangkan algoritma *random forest*.



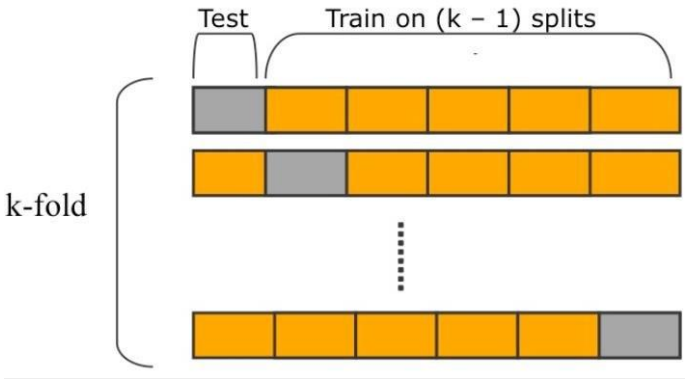
Gambar 2.2 Struktur *Random Forest*
Sumber: Spotfire

Algoritma *random forest* melakukan proses latih data dengan menggunakan lebih dari satu *decision tree* (Breiman, 2001). *Random forest* dapat mengambil keputusan untuk klasifikasi dengan menggunakan hasil *voting* dari seluruh *decision tree* yang dibuat. Semakin banyak jumlah *decision tree* yang digunakan, maka nilai

akurasi akan menjadi lebih baik. Kerangka *random forest* dapat dilihat pada Gambar 2.2.

2.1.1.2 K-*fold Cross Validation*

K-*fold cross validation* merupakan salah satu teknik yang dapat digunakan dalam evaluasi model prediksi/klasifikasi (Pandian, 2023). Dalam algoritma k-*fold cross validation*, data akan dibagi menjadi *k* bagian yang sama. Dari data yang telah dibagi, selanjutnya diambil $k - 1$ bagian untuk digunakan sebagai data latih dan 1 bagian lainnya digunakan sebagai data uji. Proses klasifikasi baik *training* maupun *testing* akan dilakukan sebanyak *k* kali dengan pengambilan data bagian yang berbeda. Dengan demikian, setiap data memiliki peluang yang sama baik dalam menjadi data latih maupun data uji. Visualisasi pembagian data pada teknik k-*fold cross validation* dapat dilihat pada Gambar 2.3.



Gambar 2.3 *K-fold Cross Validation*
Sumber: Alaoui, dkk., 2018

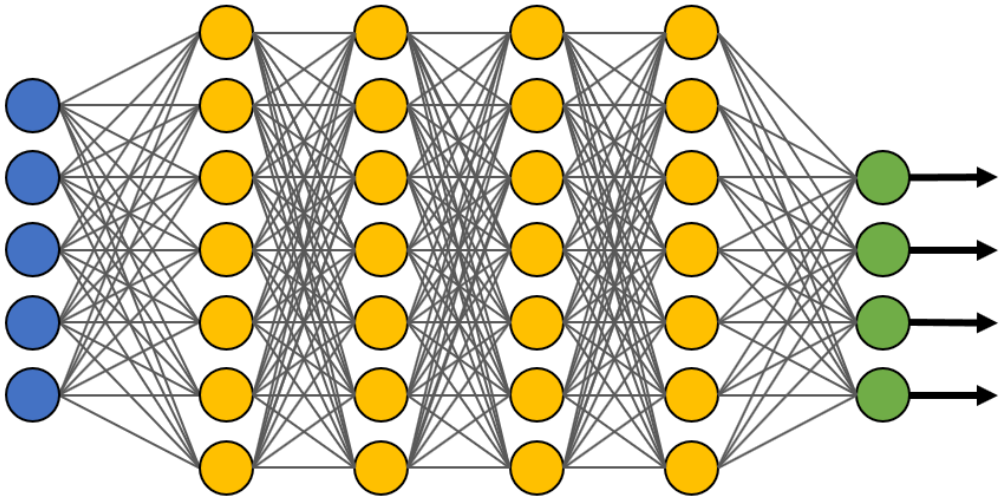
Pada penelitian ini, hasil evaluasi ditentukan berdasarkan tingkat akurasi. Adapun tingkat akurasi dapat dihitung dengan menggunakan Persamaan (2.3).

$$\text{Akurasi} = \frac{\text{Banyak Prediksi Benar}}{\text{Banyak Data Uji}} \tag{2.3}$$

Dalam k-*fold cross validation*, pada setiap percobaan akan dihitung nilai akurasinya. Hasil evaluasi akhir pada teknik ini diperoleh dari rata-rata akurasi setiap percobaan.

2.1.2 *Deep Learning*

Deep learning merupakan metode pengembangan dari *machine learning* (Setiawan, 2020). Perbedaan *deep learning* dengan *machine learning* sederhana adalah, *deep learning* melakukan proses pembelajaran yang terdiri dari beberapa lapisan (*layer*). Setiap lapisan *deep learning* akan menghasilkan fitur yang lebih kompleks dari fitur lapisan sebelumnya. Oleh karena itu, *deep learning* dapat melakukan pembelajaran mendalam dengan data besar. *Deep learning* pada umumnya sering digunakan dalam proses klasifikasi data kompleks seperti citra, suara, dan teks.



Gambar 2.4 Lapisan (*Layer*) *Deep Learning*
Sumber: Tran, 2019

Deep learning secara umum dibagi menjadi tiga lapisan utama sebagai berikut (*Amazon Web Services*, n.d).

1. Lapisan *input* (*input layer*)

   Lapisan ini merupakan lapisan awal dalam *deep learning*. Pada lapisan ini, setiap data akan dimasukkan ke dalam model *deep learning* untuk dapat diproses pada lapisan-lapisan selanjutnya.

2. Lapisan tersembunyi (*hidden layer*)

   Lapisan ini merupakan lapisan yang mengolah informasi dari data. Tujuan dari lapisan ini adalah mempelajari data sehingga dihasilkan informasi atau fitur baru. Lapisan tersembunyi biasanya dilakukan berkali-kali untuk dapat menghasilkan informasi atau fitur yang semakin dalam dari sebuah data.

3. Lapisan *output* (*ouput layer*)

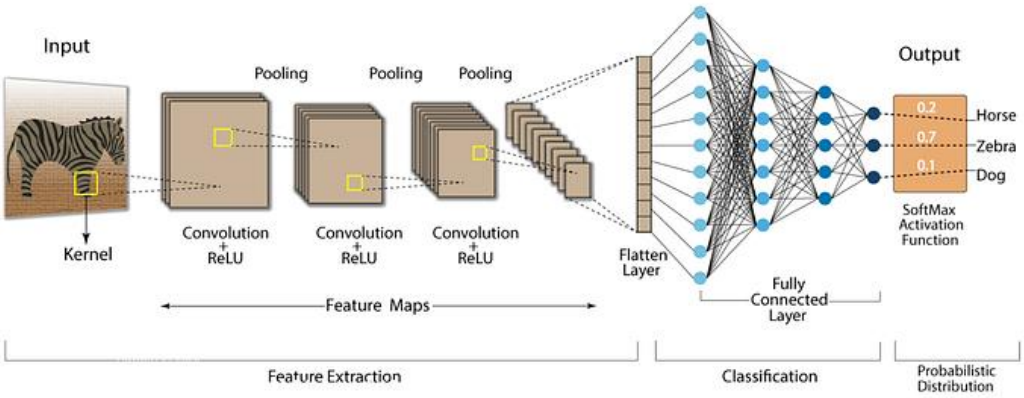
   Lapisan *output* merupakan lapisan hasil yang dapat digunakan untuk proses klasifikasi. Pada umumnya, lapisan *output* memiliki jumlah *neural* sebanyak kelas dari data yang diinginkan.

Detail pembagian lapisan pada *deep learning* dapat dilihat pada Gambar 2.4. Pada gambar tersebut terdapat tiga pembagian warna yang mewakili masing-masing lapisan pada *deep learning*. Warna biru mewakili lapisan *input*, warna kuning mewakili lapisan tersembunyi, dan warna hijau mewakili lapisan *output*.

Deep learning bekerja seperti otak manusia menggunakan jaringan saraf (*neural*). Otak manusia dapat mempelajari sesuatu dengan jutaan saraf yang saling terhubung. *Deep learning* juga dapat melakukan hal yang sama dengan membuat jaringan saraf tiruan yang saling terhubung (*fully connected*) menggunakan perhitungan matematika. Lapisan *fully connected* dalam *deep learning* berada pada lapisan *output*.

2.1.2.1 *Convolutional Neural Network* (CNN)

Convolutional Neural Network (CNN) merupakan arsitektur pengembangan dari *deep learning*. Sama halnya dengan *deep learning*, CNN juga memiliki arsitektur yang terdiri dari beberapa lapisan.



Gambar 2.5 Arsitektur *Convolutional Neural Network* (CNN)
Sumber: Ali, dkk., 2023

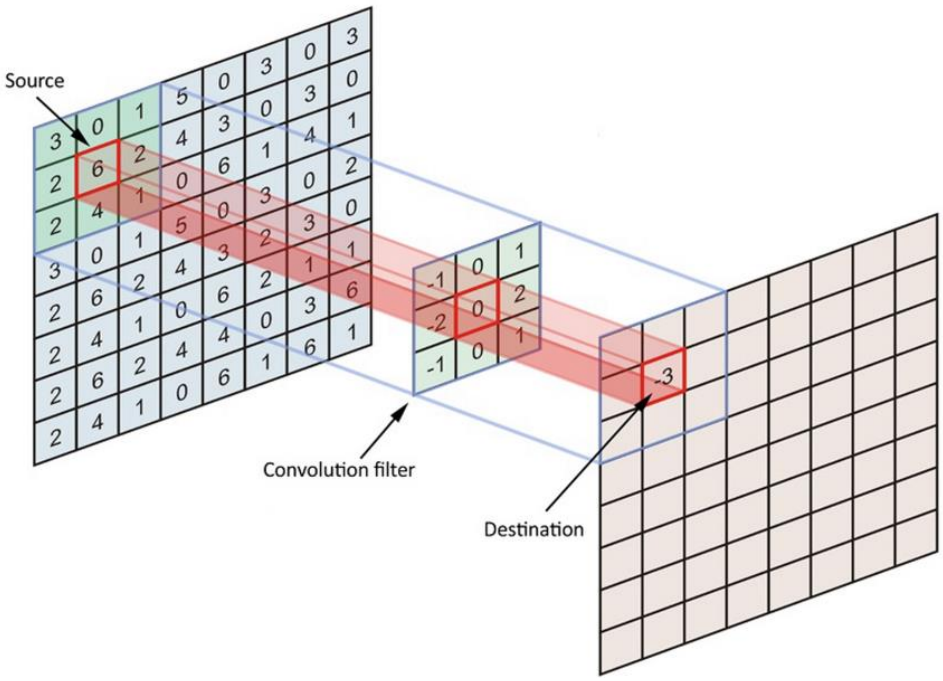
CNN memiliki dua tahapan utama yaitu tahapan ekstraksi fitur (*feature learning*) dan tahapan klasifikasi (Rahman, dkk., 2021). Arsitektur CNN dapat dilihat pada Gambar 2.5. Tahapan ekstraksi fitur terdiri dari lapisan konvolusi, dan *pooling*. Dalam lapisan konvolusi juga seringkali diterapkan fungsi aktivasi seperti ReLU (*Rectified Linear Unit*). Dalam CNN lapisan konvolusi dan *pooling* dilakukan berkali-kali untuk menghasilkan fitur yang dibutuhkan. Sedangkan tahapan klasifikasi terbagi menjadi lapisan *flatten*, *fully connected*, dan *softmax*.

Adapun penjelasan terkait masing-masing lapisan adalah sebagai berikut.

1. Lapisan konvolusi

   Pada lapisan ini, data akan diberikan *filter* khusus untuk menemukan fitur tertentu. Dalam pengolahan citra, *filter* yang digunakan berupa sebuah

matriks persegi berukuran $N \times N$. Contoh proses konvolusi pada citra dapat dilihat pada Gambar 2.6.



Gambar 2.6 Tahapan Konvolusi Citra Dalam CNN
Sumber: Llorella, dkk., 2022

Pada proses konvolusi, akan ditentukan sub-matriks berukuran $n \times n$ yang merupakan matriks *filter* yang juga disebut sebagai kernel. Kemudian akan diambil sub-matriks berukuran yang sama dari citra. Setiap elemen dalam kernel citra tersebut akan dikalikan dengan elemen pada kernel *filter* yang bersesuaian seperti terlihat pada Gambar 2.6. Setelah masing-masing elemen yang bersesuaian dikalikan selanjutnya masing-masing hasil perkalian tersebut akan dijumlahkan. Proses ini dilakukan terhadap keseluruhan citra dengan menggeser kernel secara spasial. Adapun nilai pergeseran kernel dalam *machine learning* dikenal sebagai *stride*. Secara matematis, proses konvolusi dapat dituliskan dalam bentuk Persamaan (2.4).

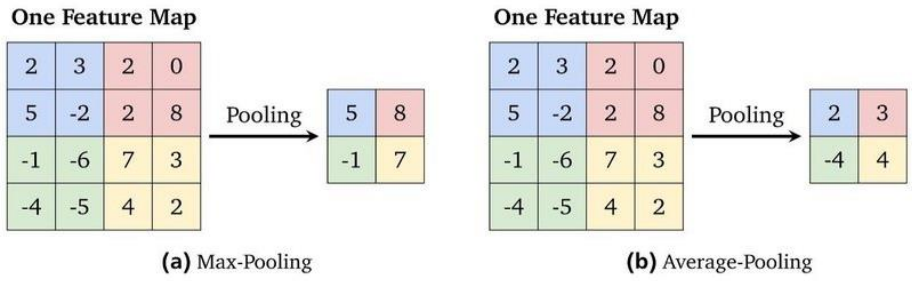
$$h(x,y) = \sum_{a=-n}^{n} \sum_{b=-n}^{n} f(a,b)\, g(x-a, y-b) \qquad (2.4)$$

Simbol * pada persamaan ini merupakan simbol konvolusi. $h(x,y)$ merupakan data *output*, $f(x,y)$ merupakan data *input*, dan $g(x,y)$ merupakan matriks *filter*, (x,y) adalah pasangan koordinat baris dan kolom pada matriks tersebut. Adapun n merupakan ukuran tinggi dan lebar kernel.

Selain itu, dalam proses konvolusi juga terdapat teknik *padding* yaitu memberi dimensi tambahan untuk keperluan konvolusi. *Padding* dilakukan untuk menjaga ukuran dimensi citra. Pada umumnya, elemen *padding* diberi nilai 0.

2. Lapisan *Pooling*

Pada lapisan ini, akan dilakukan *downsampling* atau pengurangan dimensi. *Pooling* dilakukan untuk mempersingkat waktu latih data. Pada CNN, terdapat dua jenis *pooling* yang sering digunakan yaitu *max pooling* dan *average pooling*. Perbedaan *max pooling* dan *average pooling* dapat dilihat pada Gambar 2.7.

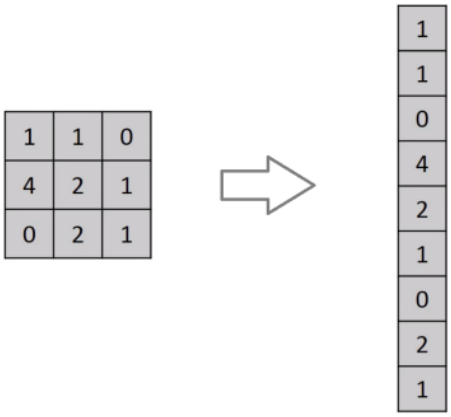


Gambar 2.7 Proses *Pooling*
Sumber: Eddine, 2019

Max pooling akan mengurangi dimensi berdasarkan nilai maksimum dari setiap jendela (kernel) berukuran $N \times N$. Sedangkan *average pooling* akan mengurangi dimensi berdasarkan nilai rata-rata dari setiap kernelnya.

3. Lapisan *Flatten*

Pada lapisan ini, data seperti citra yang merupakan data dua dimensi akan ditransformasikan menjadi vektor berukuran $N \times 1$ dimana N merupakan hasil kali dari panjang dan tinggi citra. Proses ini perlu dilakukan untuk dapat dilakukan klasifikasi. Proses *flatten* dapat dilihat pada Gambar 2.8.



Gambar 2.8 Proses *Flatten*
Sumber: Kaggle

4. Lapisan *Fully Connected*

Lapisan ini dilakukan untuk menghubungkan semua jaringan saraf yang telah dihasilkan dari proses sebelumnya. Lapisan ini mentransformasikan dimensi dari data yang telah didatarkan sebelumnya agar data dapat diklasifikasikan secara linear.

Selain empat lapisan diatas, dalam CNN juga terdapat lapisan aktivasi. Fungsi aktivasi digunakan untuk menentukan saraf mana yang perlu diaktifkan dan saraf mana yang tidak perlu diaktifkan berdasarkan bobotnya. Dalam CNN terdapat dua jenis fungsi aktivasi yang sering digunakan saat ini, yaitu fungsi dan fungsi *softmax*.

Fungsi ReLU dapat melakukan *filtering* terhadap data dengan mengubah nilai negatif menjadi 0. Bentuk fungsi ReLU secara matematis dapat dituliskan sebagai Persamaan (2.5).

$$f(x) = \max(0, x) \tag{2.5}$$

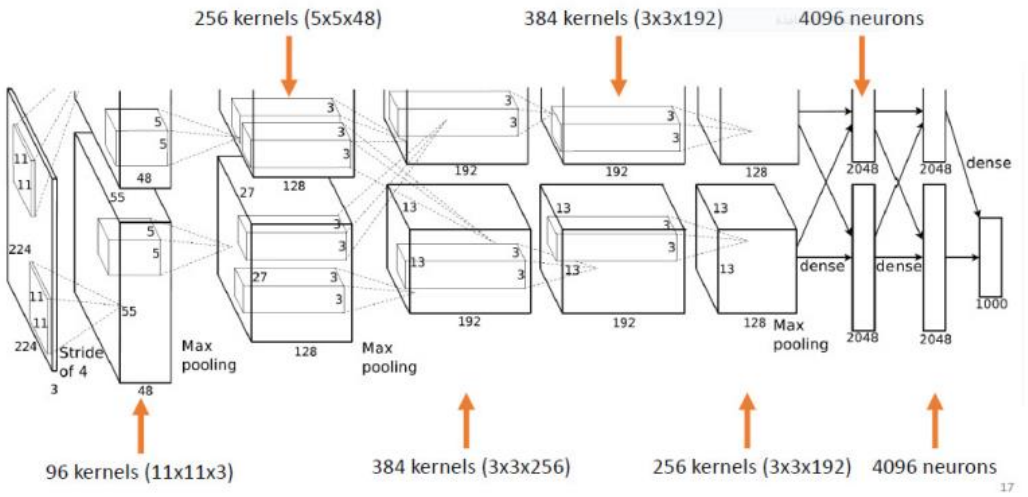
Fungsi *softmax* dilakukan untuk menyediakan probabilitas untuk setiap kelas pada *output* antara $0 - 1$. Pada umumnya, fungsi *softmax* seringkali diterapkan pada lapisan *fully connected*. Secara matematis, fungsi *softmax* dapat ditulis sebagai Persamaan (2.6).

$$\sigma(x_i) = \frac{\exp(x_i)}{\sum_j x_j} \tag{2.6}$$

2.1.2.2 *Transfer Learning*

Transfer learning merupakan teknik pendekatan *machine learning* dengan menggunakan model yang telah dilatih sebelumnya (Dharmaraj, 2022). Keunggulan dalam menggunakan *transfer learning* adalah *transfer learning* dapat mengurangi waktu yang diperlukan mesin untuk membuat model baru. Selain itu, hal ini juga sangat berguna untuk mengurangi terjadinya *error* (kesalahan) dalam proses pembelajaran.

Model *transfer learning* yang digunakan pada penelitian ini adalah model AlexNet. Model AlexNet pertama kali dikembangkan pada tahun 2012 (Yoss, 2020). Model ini telah dilatih pada kompetisi *ImageNet Large-Scale Visual Recognition Challenge* (ILSVRC) menggunakan 1,2 juta citra yang berasal dari *dataset* ImageNet dengan 1000 kelas (Krizhevsky, 2012).



Gambar 2.9 Arsitektur Model AlexNet
Sumber: Ipe, 2020

Model AlexNet memiliki delapan lapisan ekstraksi fitur dalam arsitekturnya, terdiri dari lima lapisan konvolusi dan tiga lapisan *pooling*. Pada bagian klasifikasi, model AlexNet memiliki tiga lapisan *fully connected*. Setiap lapisan dalam model AlexNet diterapkan fungsi ReLU kecuali pada lapisan *fully connected* terakhir yaitu menggunakan fungsi *softmax*. Detail arsitektur model AlexNet dapat dilihat pada Gambar 2.9 dan Tabel 2.1.

Dalam tabel tersebut, terdapat lima kolom yaitu kolom jenis lapisan yang merupakan nama lapisan (*layer*) yang akan dilakukan, kolom ukuran peta fitur yang menyajikan ukuran dimensi hasil *output* dari *layer* yang berkaitan, kolom ukuran kernel yang berisikan ukuran dimensi kernel yang akan digunakan dalam *layer* terkait, ukuran *stride* yaitu besar jarak pergeseran jendela dalam suatu *layer*, dan ukuran *padding* yang menyajikan informasi terkait besar ukuran *padding* yang digunakan pada suatu *layer*.

Tabel 2.1 Tabel Model AlexNet

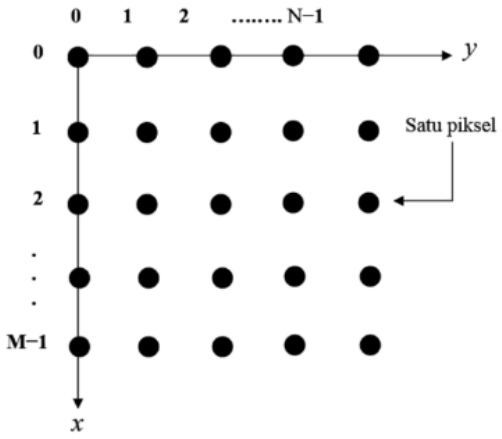
Jenis Lapisan	**Ukuran Peta Fitur**	**Ukuran Kernel**	**Ukuran *Stride***	**Ukuran *Padding***
Lapisan *input*	$227 \times 227 \times 3$ (tinggi x lebar x kanal)			
Konvolusi 1 ReLU-1	$55 \times 55 \times 96$	11×11	4×4	0×0
Max Pooling 1	$27 \times 27 \times 96$	3×3	2×2	0×0
Konvolusi 2 ReLU-2	$27 \times 27 \times 256$	5×5	1×1	2×2
Max Pooling 2	$13 \times 13 \times 256$	3×3	2×2	0×0
Konvolusi 3 ReLU-3	$13 \times 13 \times 384$	3×3	1×1	1×1
Konvolusi 4 ReLU-4	$13 \times 13 \times 384$	3×3	1×1	1×1
Konvolusi 5 ReLU-5	$13 \times 13 \times 256$	3×3	1×1	1×1
Max Pooling 5	$6 \times 6 \times 256$	3×3	2×2	0×0
Fully Connected 6 ReLU-6	4096×1 4096×1			
Fully Connected 7 ReLU-7	4096×1 4096×1			
Fully Connected 8 *Softmax*	1000×1			
Lapisan *Ouput*	1000 kelas			

Pada penelitian ini, *transfer learning* akan digunakan sebagai alat ekstraksi fitur. Untuk itu, model *transfer learning* akan dipotong sebelum lapisan *flatten*. Namun, untuk keperluan klasifikasi, pada penelitian ini model akan diambil hingga lapisan *fully connected* pertama serta menggunakan *output* pada lapisan ini sebagai vektor fitur. Dengan demikian, *output* yang akan digunakan dari model AlexNet ini adalah vektor berukuran 4096. Hal ini dilakukan untuk meningkatkan tingkat akurasi klasifikasi.

2.1.3 Citra Digital

Citra merupakan suatu gambaran yang merepresentasikan suatu objek (Andono, dkk., 2017). Citra terbagi menjadi dua jenis, yaitu citra analog dan citra digital. Citra analog merupakan citra yang tidak dapat dikelola oleh komputer. Sedangkan citra digital merupakan citra yang dapat dikelola oleh komputer. Hal ini dikarenakan citra digital terdiri dari kumpulan bit yang direpresentasikan dalam bentuk matriks M × N.

Citra digital diperoleh melalui dua tahap utama yaitu tahap *sampling* dan tahap kuantisasi. Tahap *sampling* merupakan tahap pertama yang dapat mengubah citra analog menjadi sebuah matriks berukuran M × N. Masing-masing koordinat pada matriks citra disebut sebagai piksel (*picture element*). Semakin besar ukuran matriks citra maka semakin baik hasil representasi dari citra tersebut. Matriks yang dihasilkan dari tahap *sampling* dapat dilihat pada Gambar 2.10.



Gambar 2.10 Representasi Matriks Citra Hasil Tahap *Sampling*
Sumber: Andono dkk., 2017

Tahap kuantisasi merupakan tahap yang mengubah nilai intensitas citra analog menjadi intensitas diskrit. Tahap kuantisasi akan memberikan nilai khusus pada tiap piksel yang mewakili warna citra pada tiap piksel tersebut. Nilai

intensitas/warna suatu piksel yang berada pada koordinat (x, y) dapat disimbolkan sebagai $f(x, y)$. Setelah melalui dua tahap tersebut, citra dapat direpresentasikan dalam matriks sebagai berikut.

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f\big(0,(M-1)\big) \\ f(1,0) & f(1,1) & \cdots & f\big(1,(M-1)\big) \\ \vdots & \vdots & & \vdots \\ f\big((N-1),0\big) & f\big((N-1),1\big) & \cdots & f\big((N-1),(M-1)\big) \end{bmatrix}$$

2.1.3.1 Operator *Integro-Differensial*

Operator *integro-differensial* merupakan operator yang dapat digunakan untuk mendeteksi lingkar iris mata berdasarkan perbedaan intensitas cahaya di setiap lingkar calon iris pada citra mata (Daugman, 2004). Daugman menganjurkan operator *integro-differensial* dalam penelitiannya (Hakim, 2022). Bentuk matematis operator *integro-differensial* dapat dilihat pada Persamaan (2.7).

$$\max_{(r,x_0,y_0)} \left| G_\sigma(r) * \frac{\partial}{\partial r} \oint_{s(r,x_0,y_0)} \frac{I(x,y)}{2\pi r} \, \mathrm{d}s \right| \tag{2.7}$$

dimana $I(x, y)$ merupakan intensitas pada citra mata dan s merupakan calon lingkar iris mata. Makna *integro* dalam operator ini menjelaskan bahwasanya operator ini diterapkan secara spasial terhadap calon lingkar s yang mungkin. Operator ini akan mencari nilai maksimum pada setiap hasil turunan parsial terhadap jari-jari r dari hasil integral dari kontur citra yang ternormalisasi di sepanjang lingkaran $\mathrm{d}s$ dengan titik pusat (x_0, y_0) dan jari-jari r. Fungsi $G_\sigma(r)$ yaitu fungsi *Gaussian smoothing* dengan skala σ diterapkan untuk menghilangkan *noise* pada citra untuk memaksimalkan hasil deteksi. Fungsi *Gaussian* dapat dituliskan dalam bentuk Persamaan (2.8).

$$G_\sigma(r) = \frac{1}{\sigma\sqrt{2\pi}}\, exp\left(-\frac{(r-r_0)^2}{2\sigma^2}\right) \tag{2.8}$$

Pencarian ini dilakukan secara iteratif dengan memvariasikan tiga parameter utama terhadap jari-jari dan titik pusat (r, x_0, y_0). Hasil dari operator ini akan mengembalikan lingkaran dengan titik pusat (x_0, y_0) dan jari-jari r yang memiliki nilai hasil turunan integral terhadap kontur dengan skala *blur* sebesar σ yang maksimum.

2.1.3.2 *Masking*

Masking merupakan metode yang dapat digunakan untuk segmentasi iris. Segmentasi iris merupakan proses pemisahan bagian iris pada citra untuk dapat dianalisis. Dengan memisahkan iris dari bagian lain pada gambar mata, maka pengenalan iris dapat dilakukan dengan lebih akurat.

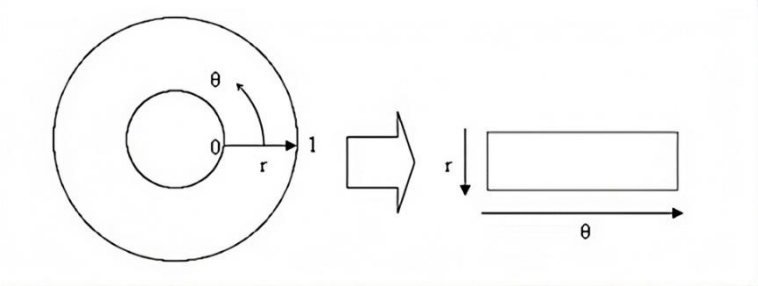
Masking diterapkan terhadap citra yang sudah dilokalisasi dengan menggunakan operator *integro-differensial*. Pada proses ini, dibuat citra biner sebagai *template* area citra yang perlu disegmentasi. Citra ini akan berisi nilai biner yaitu 0 dan 1 dimana area iris akan berada pada piksel yang bernilai 1. Proses *masking* dilakukan dengan mengisi setiap piksel pada citra biner yang bernilai 1 dengan citra iris yang memiliki koordinat yang sama. Secara matematis proses *masking* dapat ditulis dalam bentuk Persamaan (2.9).

$$h(x,y) = f(x,y)g(x,y) \tag{2.9}$$

dimana $h(x,y)$, $f(x,y)$, dan $g(x,y)$ berturut-turut merupakan citra hasil *masking*, citra asli, dan citra biner dan (x,y) merupakan koordinat setiap piksel dalam citra.

2.1.3.3 *Rubber Sheet*

Proses normalisasi iris dilakukan dengan menggunakan metode *rubber sheet*. Normalisasi iris diperlukan untuk mengurangi *noise* yang berasal dari wilayah hitam di sekitar lingkaran yang dapat mempengaruhi fitur. Proses normalisasi diperlukan untuk membantu memaksimalkan proses klasifikasi. Selain itu, pada penelitian ini normalisasi juga diperlukan supaya fitur dapat diekstraksi menggunakan metode *transfer learning*. Visualisasi proses normalisasi iris dapat dilihat pada Gambar 2.11.



Gambar 2.11 Normalisasi Iris Metode *Rubber Sheet*
Sumber: Rezika, 2018

Pada algoritma *rubber sheet*, setiap piksel pada area iris akan diakses menggunakan koordinat polar (r, θ) dengan jari-jari r dan sudut θ untuk selanjutnya diubah menjadi bentuk persegi. Adapun nilai setiap piksel pada citra hasil normalisasi dapat dicari pada citra iris mata dengan menggunakan Persamaan (2.10), Persamaan (2.11) dan Persamaan (2.12).

$$I\big(x(r,\theta), y(r,\theta)\big) \rightarrow I(r,\theta) \tag{2.10}$$

$$x(r,\theta) = (1-r)x_p(\theta) + rx_i(\theta) \tag{2.11}$$

$$y(r,\theta) = (1-r)y_p(\theta) + ry_i(\theta) \tag{2.12}$$

$I(x, y)$ merupakan area iris dalam koordinat kartesius. (r, θ) merupakan koordinat polar yang diperoleh dari koordinat kartesius di antara koordinat lingkar

pupil x_p, y_p dan koordinat lingkar iris x_i, y_i sepanjang sudut θ. Adapun koordinat lingkar pupil x_p, y_p dan koordinat lingkar iris x_i, y_i pada sudut θ berturut-turut dapat dicari menggunakan Persamaan (2.13), Persamaan (2.14), Persamaan (2.15), dan Persamaan (2.16).

$$x_p(\theta) = x_0 + r_p \cos\theta \tag{2.13}$$

$$y_p(\theta) = y_0 + r_p \sin\theta \tag{2.14}$$

$$x_i(\theta) = x_0 + r_i \cos\theta \tag{2.15}$$

$$y_i(\theta) = y_0 + r_i \sin\theta \tag{2.16}$$

dimana x_0, y_0 merupakan titik pusat lingkar pupil dan lingkar iris.

Selanjutnya masing-masing piksel yang telah diambil dalam bentuk koordinat polar akan digunakan untuk menyusun citra baru yang berbentuk persegi. Dalam penelitian ini, ukuran persegi yang digunakan adalah 227×227 piksel untuk menyesuaikan model *transfer learning* yang digunakan. Adapun nilai r dan θ yang digunakan untuk merepresentasikan citra hasil normalisasi dengan koordinat (n, m) berturut-turut dapat ditulis dalam Persamaan (2.17) dan Persamaan (2.18).

$$r_m = \frac{m}{227} \quad , m = \{0, 1, 2, 3, \dots, 226\} \tag{2.17}$$

$$\theta_n = 360\frac{n}{227} \quad , n = \{0, 1, 2, 3, \dots, 226\} \tag{2.18}$$

2.1.3.4 *Image Enhancement*

Image enhancement (peningkatan citra) merupakan proses perbaikan atau peningkatan kualitas citra (Petrou, 2010). *Image enhancement* diterapkan terhadap citra untuk menghilangkan *noise* atau meningkatkan kontras citra.

Salah satu jenis *image enhancement* yang dapat digunakan adalah *histogram equalization*. *Histogram equalization* merupakan proses perataan intensitas cahaya pada citra (Sudhakar, 2017). *Histogram equalization* dilakukan dengan mengubah nilai setiap piksel dengan menggunakan nilai distribusi kumulatif. Fungsi distribusi kumulatif (*cumulative distribution function*) dapat ditulis sebagai Persamaan (2.19).

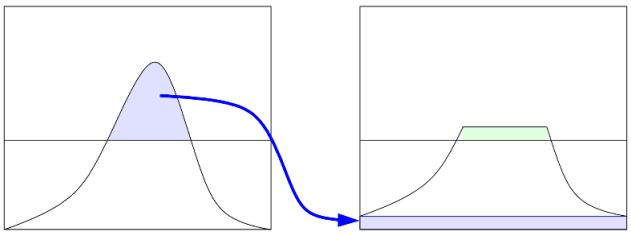
$$cdf(v) = \sum_{i=0}^{v} n_i \, , 0 \leq v \leq L \tag{2.19}$$

dimana n_i merupakan banyak piksel dengan intensitas i pada citra dan n merupakan banyak keseluruhan piksel pada citra.

Secara matematis, *histogram equalization* dapat ditulis sebagai Persamaan (2.20).

$$h(v) = round\left(\frac{cdf(v) - cdf_{min}}{n - cdf_{min}} \times (L-1)\right) \tag{2.20}$$

dimana $h(v)$ merupakan nilai hasil ekualisasi citra dengan intensitas v, $cdf(v)$ merupakan nilai distribusi kumulatif terhadap intensitas v, cdf_{min} merupakan nilai distribusi kumulatif minimum dan L merupakan banyak nilai intensitas yang mungkin pada citra (biasanya 256).



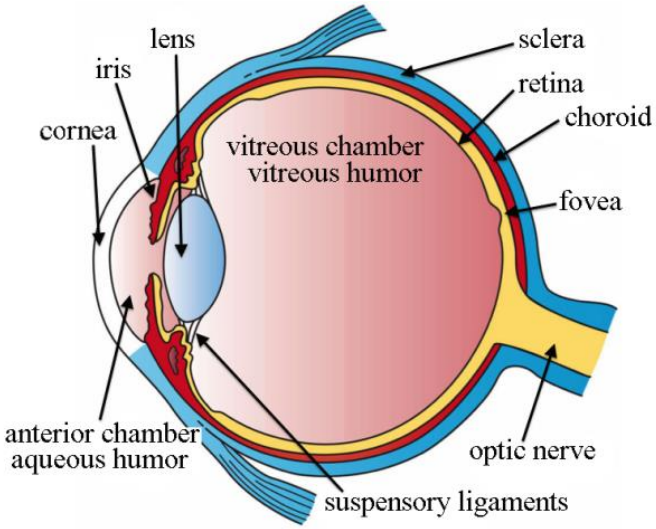
Gambar 2.12 Penerapan *Clip Limit* Pada CLAHE
Sumber: Vinay, 2021

Pada penelitian ini, algoritma yang akan digunakan merupakan algoritmma pengembangan dari *histogram equalization* yaitu *contrast limited adaptive*

histogram equalization (CLAHE). Pada CLAHE, citra akan dibagi menjadi beberapa jendela kecil (*tile*) dengan ukuran tertentu. Pada setiap *tile* akan dilakukan *histogram equalization* secara lokal. Setelah proses *histogram equalization*, selanjutnya kontras pada setiap jendela akan dipotong dengan batas tertentu. Batas kontras pada CLAHE disebut sebagai *clip limit* (Pizer, 1987). Jumlah kontras yang telah dipotong tersebut akan disebarkan kembali seperti terlihat pada Gambar 2.12.

2.1.4 Iris

Iris mata adalah organ bagian dari mata yang berada di antara pupil dan sklera (bagian putih pada mata). Detail letak iris mata dapat dilihat pada Gambar 2.13. Iris mata memiliki bentuk seperti gelang. Pada mata manusia, iris mata merupakan otot yang berfungsi untuk melebarkan atau mengecilkan pupil untuk mengatur jumlah intensitas cahaya yang masuk ke retina.



Gambar 2.13 Struktur Anatomi Mata
Sumber: FSCJ

Iris mata merupakan organ tubuh yang memiliki pola kompleks dan unik (Patel, 2008). Peluang dua iris mata memiliki pola yang sama adalah sekitar 1 dari

10^{78}. Selain itu, iris mata merupakan organ tubuh yang bersifat internal. Iris mata dilindungi oleh kornea. Hal ini mengakibatkan iris mata menjadi organ tubuh yang aman dan tidak mudah berubah seiring berjalannya waktu (Hakim, 2022). Oleh karena itu, iris mata merupakan organ tubuh manusia yang sangat baik digunakan dalam biometrik dibandingkan dengan organ tubuh lainnya.

2.2 Kajian Integrasi Topik dengan Al-Qur'an/Hadits

Pada dasarnya, segala sesuatu dalam alam semesta ini diciptakan oleh Allah SWT dengan sebaik-baiknya. Setiap hal yang tercipta di dunia ini memiliki ukuran dan karakteristiknya masing-masing. Kesempurnaan ciptaan Allah dengan segala perhitungannya tertulis pada Al-Qur'an surat Al-Qamar ayat 49 yang artinya (Kemenag RI, 2022):

"Sesungguhnya Kami menciptakan segala sesuatu menurut ukuran" (QS. Al-Qamar:4)

Dalam ayat tersebut, disebutkan bahwa Allah SWT menciptakan segala sesuatu baik yang ada di langit maupun di bumi sesuai dengan ukurannya. Salah satu contoh dan bukti dari hal tersebut dapat terlihat pada iris mata manusia. Setiap manusia, memiliki karakteristik iris mata yang berbeda.

Seiring perkembangan teknologi, hal ini dapat dibuktikan dengan ilmu komputasi seperti *image processing*. Dengan menggunakan *image processing*, ditemukan bahwa setiap manusia memiliki iris mata dengan karakteristik dalam bentuk vektor dengan angka yang berbeda. Hal ini sangat bersesuaian dengan yang tertulis dalam surat Al-Qamar ayat 49. Dengan demikian, penelitian ini secara tidak langsung dapat menjadi media penambah wawasan sekaligus bukti akan kebenaran dari kalam Allah SWT.

2.3 Kajian Topik dengan Teori Pendukung

Penelitian ini dilakukan untuk mengembangkan pengetahuan dalam bidang identifikasi biometrik terutama iris mata. Pada era ini, identifikasi biometrik merupakan salah satu topik dan bidang keilmuan yang banyak dibutuhkan untuk keperluan keamanan data. Pemilihan iris mata sebagai objek penelitian adalah dikarenakan iris mata merupakan salah satu organ tubuh bagian dalam. Dibandingkan dengan organ lainnya, iris mata cenderung memiliki pola yang tetap setiap tahunnya. Iris mata hampir tidak mengalami perubahan dari waktu ke waktu. Hal ini membuat iris mata menjadi salah satu objek biometrik yang kuat untuk keamanan data.

Pada penelitian ini, proses identifikasi iris mata dibagi menjadi beberapa tahapan yaitu tahap *preprocessing*, ekstraksi fitur, dan tahap klasifikasi. Pada tahap *preprocessing*, citra iris mata akan disegmentasi untuk mendapatkan area iris mata yang diinginkan menggunakan algoritma yang dikenalkan oleh Daugman. Setelah proses segmentasi, selanjutnya citra akan dinormalisasi menggunakan metode *rubber sheet* untuk mendapatkan pola iris berbentuk persegi. Setelah iris dinormalisasi, selanjutnya dilakukan *image enhancement* untuk meningkatkan kualitas citra iris sekaligus untuk membuat pola iris terlihat semakin tegas. Tahap *preprocessing* diperlukan untuk mempersiapkan citra menjadi lebih matang untuk diklasifikasikan.

Pada tahap ekstraksi fitur, digunakan metode *transfer learning* yang merupakan model *deep learning convolutional neural network* yang telah dilatih sebelumnya. Hal ini dilakukan untuk mendapatkan hasil ekstraksi terbaik tanpa perlu mengembangkan model baru. Model AlexNet merupakan model yang

digunakan untuk ekstraksi fitur pada penelitian ini. Model AlexNet telah dilatih untuk mengklasifikasikan 1,2 juta citra ke dalam 1000 model dari *dataset* ImageNet. Penggunaan model AlexNet dipilih untuk mengurangi terjadinya kesalahan dalam *machine learning*.

Selanjutnya, proses klasifikasi dilakukan dengan menggunakan algoritma *random forest*. Algoritma *random forest* dipilih karena memiliki tingkat akurasi yang cukup tinggi dalam menangani data berukuran besar. Di samping itu, hal ini juga dilakukan untuk mendapatkan wawasan baru yang belum diperoleh dari penelitian sebelumnya. Pada penelitian sebelumnya, proses klasifikasi dilakukan dengan menggunakan algoritma *support vector machine* (SVM). Dengan demikian, tujuan dari penelitian ini adalah untuk mengetahui tingkat akurasi yang dapat diperoleh dalam identifikasi citra iris mata menggunakan metode *transfer learning* dan *random forest*.

BAB III

METODE PENELITIAN

3.1 Jenis Penelitian

Jenis penelitian yang digunakan pada penelitian ini adalah penelitian kuantitatif dengan metode eksperimen. Dengan demikian, penelitian ini dilakukan untuk menguji efektif atau tidaknya variabel eksperimen. Hasil yang digunakan untuk menguji pada penelitian ini adalah hasil berupa tingkat akurasi implementasi metode *transfer learning* dan *random forest* pada klasifikasi citra iris mata.

3.2 Data dan Sumber Data

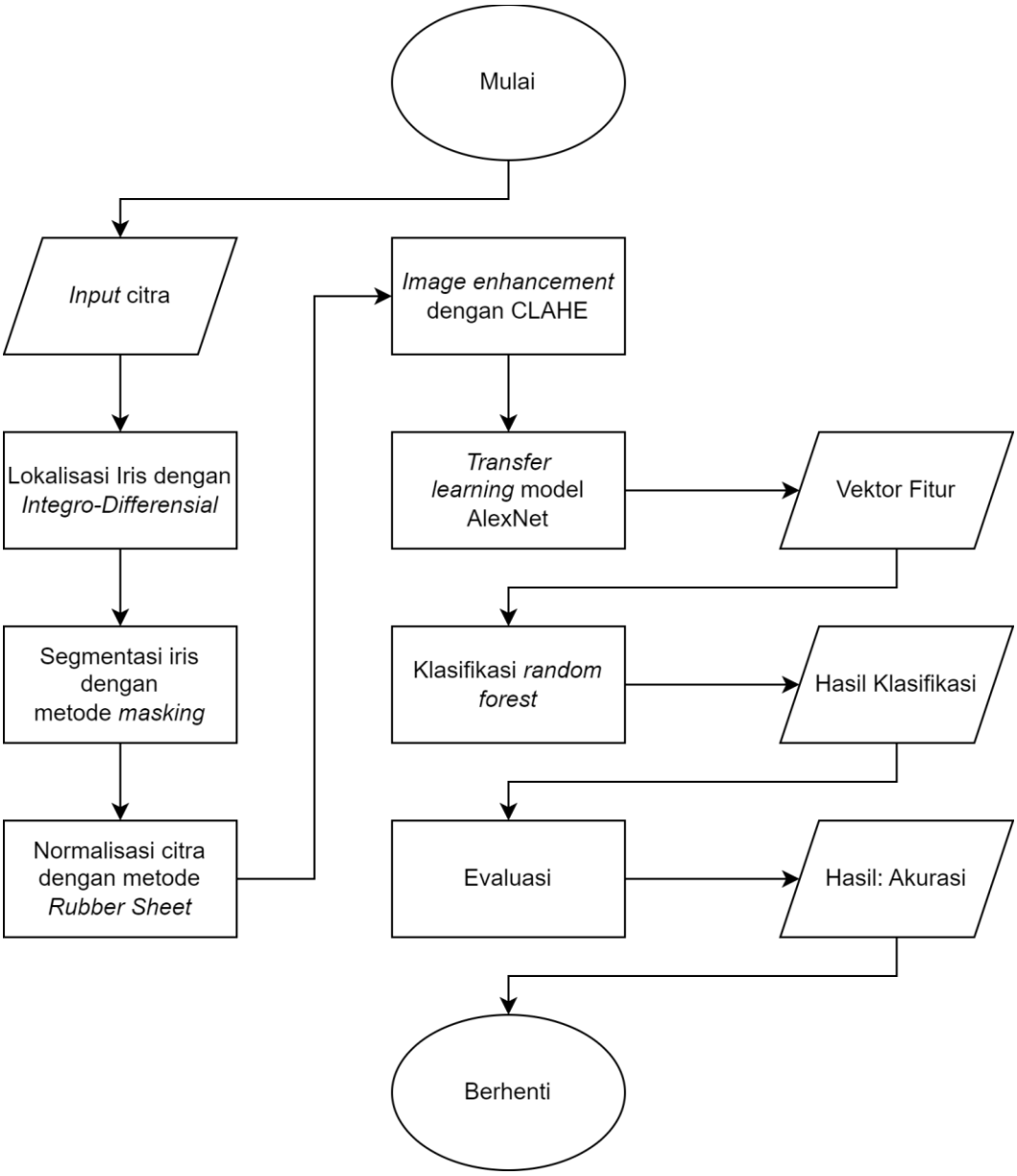
Data yang digunakan pada penelitian ini yakni data berupa citra iris mata. Citra iris mata pada penelitian ini diambil dari *dataset* CASIA V1. *Dataset* CASIA V1 terdiri dari 756 citra yang terbagi menjadi 108 kelas. Setiap citra pada *dataset* ini merupakan citra hitam putih (*grayscale*) dengan dimensi 320×280 piksel dan ekstensi *bmp*. (http://biometrics.idealtest.org/#/datasetDetail/1)

3.3 Tahapan Penelitian

Adapun tahapan penelitian seperti ditunjukkan pada Gambar 3.1 dapat dibagi menjadi dua bagian, yaitu implementasi dan evaluasi. Adapun tahapan implementasi dapat diuraikan sebagai berikut.

1. Segmentasi Citra

   Pada tahap segmentasi, terdapat beberapa proses sebagai berikut.



Gambar 3.1 Diagram Alir Penelitian

a. Lokalisasi batas lingkar iris mata dengan operator *integro-differensial* menggunakan Persamaan (2.7)

b. *Masking* dengan cara mengalikan citra asli dengan citra biner hasil lokalisasi citra dengan menggunakan Persamaan (2.9)

2. Normalisasi Citra

Tahap normalisasi citra dilakukan melalui beberapa proses berikut.

a. Mencari koordinat polar dari citra iris mata dengan menggunakan Persamaan (2.17) dan Persamaan (2.18)
b. Menentukan koordinat pada lingkar pupil dan lingkar iris relatif terhadap koordinat polar dengan menggunakan Persamaan (2.13), Persamaan (2.14), Persamaan (2.15) dan Persamaan (2.16)
c. Mensubstitusikan nilai piksel pada citra iris mata terhadap piksel yang relatif pada citra hasil normalisasi menggunakan Persamaan (2.10), Persamaan (2.11) dan Persamaan (2.12)

3. *Image Enhancement*

Pada penelitian ini, proses *image enhancement* dilakukan melalui beberapa tahap berikut.

a. Menentukan nilai *clip limit* dan ukuran *tile* yang akan digunakan dalam proses CLAHE. Dalam penelitian ini, penulis memilih nilai *clip limit* sebesar 4 dengan ukuran *tile* 8×8 piksel
b. Melakukan *histogram equalization* pada setiap *tile* atau jendela secara spasial menggunakan Persamaan (2.20)
c. Menerapkan *clip limit* terhadap seluruh piksel pada citra

4. Melakukan tahap ekstraksi fitur menggunakan metode *transfer learning* dengan model AlexNet yang telah tertulis pada Tabel 2.1 sehingga menghasilkan vektor fitur berukuran 4096×1
5. Klasifikasi Citra

Adapun proses klasifikasi citra dilakukan melalui beberapa tahap berikut.

a. Melakukan proses *training* data untuk menghasilkan model *random forest* dengan membuat 300 *decision tree* menggunakan Persamaan (2.2)

b. Melakukan proses *testing* menggunakan model *random forest* yang telah dibuat

Sedangkan proses evaluasi pada penelitian ini dilakukan berdasarkan tahapan berikut.

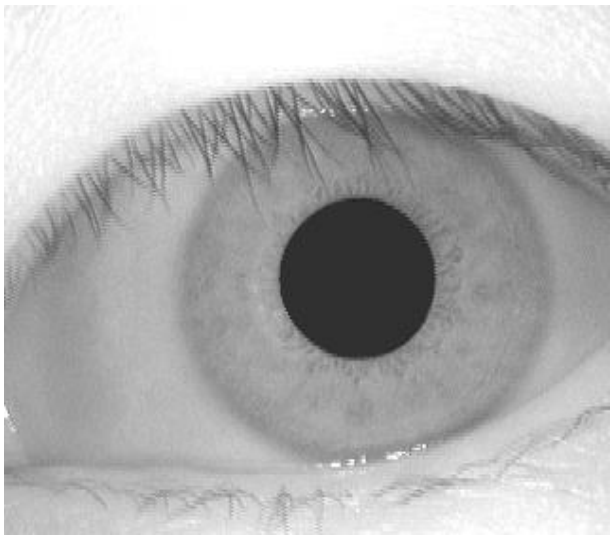
1. Membagi data menjadi k bagian yang sama. Dalam penelitian ini, penulis memilih $k = 6$ sehingga setiap bagian menghimpun 126 data
2. Melakukan proses evaluasi menggunakan k-*fold cross validation* serta menghitung nilai akurasi pada setiap percobaan menggunakan Persamaan (2.3)
3. Menghitung rata-rata nilai akurasi pada setiap percobaan k-*fold cross validation* sehingga diperoleh nilai akurasi akhir.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Segmentasi Iris

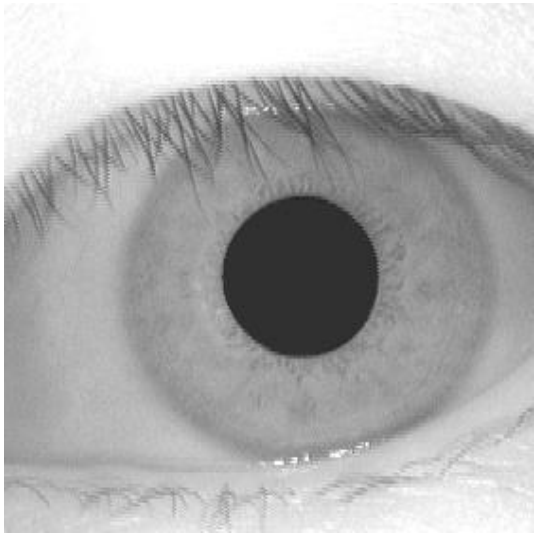
Pada proses segmentasi dilakukan dua tahapan utama yaitu lokalisasi iris menggunakan operator *integro-differensial* dan *masking*. Proses lokalisasi iris dilakukan untuk menentukan lingkar iris mata. Dalam penelitian ini, operator *integro-differensial* digunakan untuk menentukan lingkar pupil. Lingkar pupil dipilih dikarenakan pupil merupakan bagian yang terdekat dengan iris serta memiliki titik pusat lingkaran yang sama dengan iris mata. Sebagai sampel penelitian, digunakan salah satu citra iris mata seperti terlihat pada Gambar 4.1.



Gambar 4.1 Sampel Citra Iris Mata

Untuk mempermudah pencarian lingkar pupil, citra iris mata perlu dipotong terlebih dahulu sehingga diperoleh citra persegi dengan ukuran lebar dan tinggi yang sama. Pada penelitian ini, citra iris mata memiliki ukuran dimensi 320×280 piksel. Untuk memperoleh citra persegi, pada penelitian ini citra akan dipotong

sepanjang 30 piksel dari sisi kiri dan 10 piksel dari sisi kanan sehingga lebar citra akan berkurang sebanyak 40 piksel. Dengan demikian, diperoleh citra persegi dengan ukuran dimensi 280×280 piksel seperti terlihat pada Gambar 4.2.



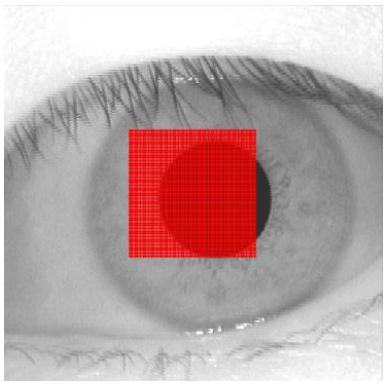
Gambar 4.2 Citra Iris Mata Yang Telah Dipotong

Adapun bentuk matriks dari citra iris mata tersebut adalah sebagai berikut (selengkapnya dapat dilihat pada Lampiran 1).

$$\begin{bmatrix} 255 & 255 & 255 & \cdots & 241 \\ 255 & 247 & 249 & \cdots & 243 \\ 253 & 249 & 247 & \cdots & 241 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 205 & 201 & 193 & \cdots & 245 \end{bmatrix}$$

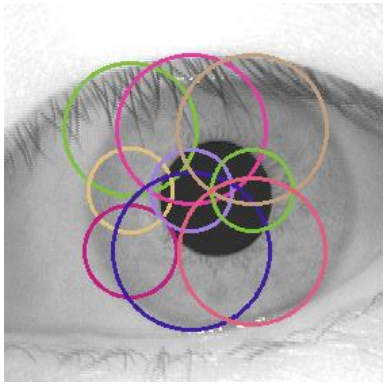
Operator *integro-differensial* mencari lingkaran dengan nilai intensitas maksimum dari seluruh lingkaran calon pupil yang mungkin pada citra iris mata. Operastor *integro-differensial* melakukan iterasi terhadap setiap lingkaran calon iris mata (r, x_0, y_0) yang mungkin pada citra iris mata dimana setiap lingkaran akan dihitung nilai intensitasnya menggunakan Persamaan (2.7). Untuk mengurangi biaya komputasi pada proses lokalisasi, pada penelitian ini pemilihan koordinat

calon titik pusat pupil dibatasi dengan jarak antar koordinat terpilih sebesar 2 piksel. Pemilihan koordinat calon titik pusat pupil dapat dilihat pada Gambar 4.3.



Gambar 4.3 Pemilihan Koordinat Calon Titik Pusat Pupil

Batas yang digunakan dalam pemilihan koordinat tersebut yaitu $\frac{1}{3}$ lebar dari kiri dan dari kanan serta $\frac{1}{3}$ tinggi dari atas dan dari bawah. Selain itu, nilai jari-jari yang mungkin pada penelitian ini juga dibatasi yaitu antara $30 - 60$ piksel. Dengan demikian, operator *integro-differensial* akan digunakan untuk menghitung nilai intensitas dari setiap lingkaran yang terpilih seperti terlihat pada Gambar 4.4.



Gambar 4.4 Lingkaran Calon Pupil Pada Citra Iris Mata

Sebagai contoh, digunakan lingkaran dengan koordinat titik pusat $(93, 93)$ dengan jari-jari sebesar 30 piksel. Berdasarkan matriks citra sampel di atas, diperoleh hasil perhitungan operator *integro-differensial* serta konvolusi dengan

fungsi *Gaussian smoothing* pada Persamaan (2.8) dengan kernel berukuran $(1, 5)$ dan skala $\sigma = 5$ pada lingkaran tersebut sebagaimana pada Persamaan (4.1) (elemen piksel pada lingkar calon pupil dengan titik pusat $(93, 93)$ dan jari-jari 30 piksel selengkapnya dapat dilihat pada Lampiran 2).

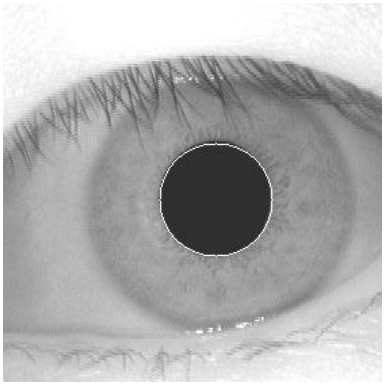
$$G_\sigma(30) * \frac{\partial}{\partial r} \oint_{s(30,93,93)} \frac{I(x,y)}{2\pi r} \, \mathrm{d}s = 0{,}155 \tag{4.1}$$

Perhitungan tersebut selanjutnya dilakukan kembali terhadap seluruh nilai jari-jari yang mungkin yaitu antara $30 - 60$ piksel

Proses tersebut selanjutnya dilakukan terhadap seluruh koordinat calon titik pusat pupil yang mungkin sehingga diperoleh hasil keseluruhan proses operator *integro-differensial* pada Persamaan (4.2) (hasil *integro-differensial* masing-masing calon lingkar iris dapat dilihat pada Lampiran 3).

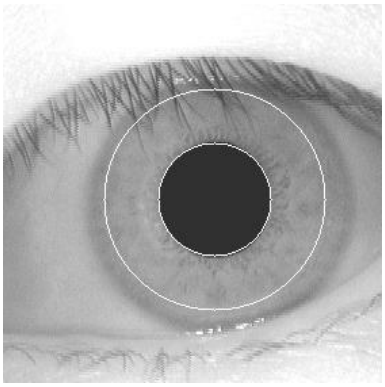
$$\max_{(r,x_0,y_0)} \left| G_\sigma(r) * \frac{\partial}{\partial r} \oint_{s(r,x_0,y_0)} \frac{I(x,y)}{2\pi r} \, \mathrm{d}s \right| = 19{,}619 \tag{4.2}$$

Dengan demikian, diperoleh lingkar pupil berada pada lingkaran yang memiliki nilai intensitas maksimum yaitu lingkaran dengan koordinat titik pusat $(157, 145)$ dengan jari-jari sebesar 42 seperti terlihat pada Gambar 4.5.



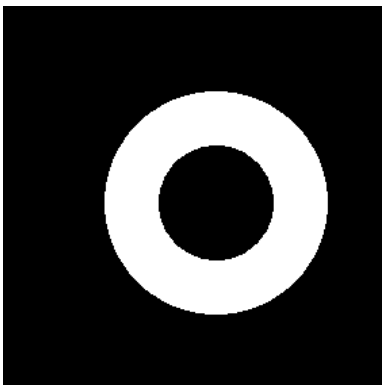
Gambar 4.5 Hasil *Integro-Differensial*

Selanjutnya untuk memudahkan pencarian lingkar iris, pada penelitian ini peneliti menggunakan titik pusat pupil mengingat iris dan pupil cenderung memiliki titik pusat yang sama. Jari-jari iris diambil dari jari-jari pupil ditambah 40 sehingga diperoleh lingkar iris yaitu lingkaran dengan koordinat titik pusat $(157, 145)$ dengan jari-jari sebesar 82 seperti terlihat pada Gambar 4.6.



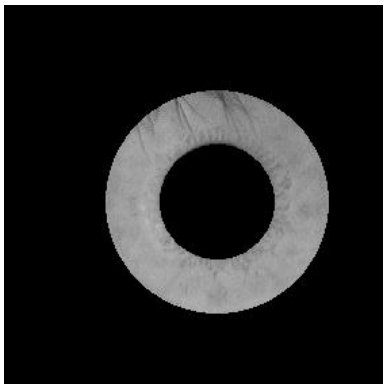
Gambar 4.6 Hasil Lokalisasi Iris Mata

Untuk melakukan proses segmentasi, tahap selanjutnya setelah lokalisasi citra iris mata adalah *masking*. Untuk melakukan *masking* maka diperlukan citra biner sebagai *template masking*. Citra biner diambil dengan mengubah nilai setiap piksel yang berada di antara batas lingkar pupil dan batas lingkar iris dengan 1 sedangkan piksel lainnya diubah menjadi 0. Citra biner tersebut dapat dilihat pada Gambar 4.7.



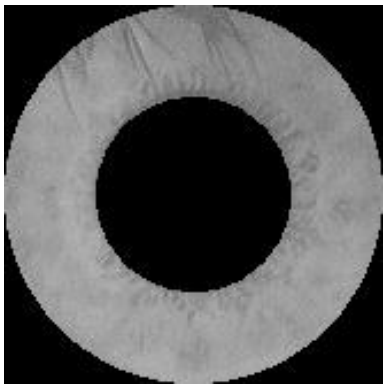
Gambar 4.7 Citra *Template* Untuk Proses *Masking*

Setelah diperoleh citra biner tersebut, selanjutnya proses segmentasi dapat dilakukan dengan mengalikan citra asli dengan citra biner menggunakan Persamaan (2.9). Dengan demikian, diperoleh hasil segmentasi citra iris mata seperti terlihat pada Gambar 4.8.



Gambar 4.8 Hasil Segmentasi Iris Mata

Untuk memudahkan proses selanjutnya, yaitu normalisasi iris maka wilayah yang bernilai 0 pada citra hasil segmentasi akan dipangkas sehingga diperoleh citra seperti pada Gambar 4.9 (matriks hasil segmentasi iris dapat dilihat pada Lampiran 4).



Gambar 4.9 Hasil Segmentasi Iris Yang Telah Dipangkas

4.2 Normalisasi Iris

Pada penelitian ini, proses normalisasi dilakukan untuk mengubah bentuk iris mata dari lingkaran menjadi persegi. Hal ini dilakukan untuk mengurangi banyak piksel selain iris mata untuk memaksimalkan hasil ekstraksi fitur. Berdasarkan Gambar 2.11 hasil normalisasi iris mata merupakan citra persegi dengan panjang θ dan tinggi r dimana (r, θ) merepresentasikan koordinat polar suatu piksel iris mata.

Pada penelitian ini, ukuran citra hasil normalisasi diatur sehingga mendapatkan citra persegi berukuran 227×227. Hal ini dilakukan untuk menyesuaikan ukuran *input* pada proses ekstraksi fitur menggunakan *transfer learning*. Sebagai contoh berdasarkan citra sampel, diketahui koordinat titik pusat lingkar pupil dan lingkar iris adalah $(x_0, y_0) = (95,95)$ dengan jari-jari pupil $r_p = 55$ dan jari-jari iris $r_i = 95$. Dengan menggunakan Persamaan (2.17) dan Persamaan (2.18) dapat diperoleh koordinat polar (r_m, θ_n) dengan $m = 100$ dan $n = 100$ seperti pada Persamaan (4.3) dan Persamaan (4.4).

$$r_m = \frac{m}{227}$$
$$r_{100} = \frac{100}{227} = 0{,}44 \tag{4.3}$$

$$\theta_n = 360\frac{n}{227}$$
$$\theta_{100} = 360\frac{100}{227} = 158{,}4 \tag{4.4}$$

Dengan demikian, dapat diperoleh koordinat lingkar pupil x_p, y_p dan koordinat lingkar iris x_i, y_i pada sudut θ dengan menggunakan Persamaan (2.13), Persamaan (2.14), Persamaan (2.15), dan Persamaan (2.16) berturut-turut pada Persamaan (4.5), Persamaan (4.6), Persamaan (4.7), dan Persamaan (4.8).

$$x_p(\theta) = x_0 + r_p \cos\theta$$
$$x_p(158{,}4^\circ) = 95 + 55\cos 158{,}4^\circ = 43 \quad (4.5)$$

$$y_p(\theta) = y_0 + r_p \sin\theta$$
$$y_p(158{,}4^\circ) = 95 + 55\sin 158{,}4^\circ = 115{,}24 \quad (4.6)$$

$$x_i(\theta) = x_0 + r_i \cos\theta$$
$$x_i(158{,}4^\circ) = 95 + 95\cos 158{,}4^\circ = 6{,}67 \quad (4.7)$$

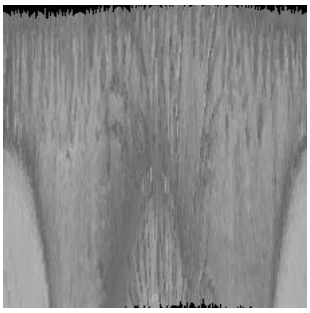
$$y_i(\theta) = y_0 + r_i \cos\theta$$
$$y_i(158{,}4^\circ) = 95 + 95\sin 158{,}4^\circ = 129{,}97 \quad (4.8)$$

Selanjutnya, dengan menggunakan Persamaan (2.11) dan Persamaan (2.12) diperoleh koordinat kartesius (x, y) dari koordinat polar $(0{,}0)$ sebagaimana Persamaan (4.9) dan Persamaan (4.10).

$$x(r,\theta) = (1-r)x_p(\theta) + rx_i(\theta)$$
$$x(0{,}44; 158{,}4^\circ) = (1-0{,}44)\, x_p(158{,}4^\circ) + 0{,}44\, x_i(158{,}4^\circ) = 27{,}01 \quad (4.9)$$

$$x(r,\theta) = (1-r)x_p(\theta) + rx_i(\theta)$$
$$y(0{,}44; 158{,}4^\circ) = (1-0{,}44)\, y_p(158{,}4^\circ) + 0{,}44\, y_i(158{,}4^\circ) = 121{,}72 \quad (4.10)$$

Berdasarkan data citra sampel, diketahui nilai intensitas piksel pada titik koordinat $(27, 121)$ adalah 118. Dengan demikian, diperoleh intensitas piksel pada citra hasil normalisasi dengan koordinat $(100, 100)$ adalah 118. Langkah ini diulangi secara berulang sehingga diperoleh citra hasil normalisasi berukuran 227×227 piksel seperti pada Gambar 4.10 (matriks hasil normalisasi iris dapat dilihat pada Lampiran 5).



Gambar 4.10 Hasil Normalisasi Iris Mata

4.3 *Image Enhancement*

Proses *image enhancement* pada penelitian ini dilakukan dengan menggunakan metode *contrast limited adaptive histogram equalization* (CLAHE). Nilai *clip limit* yang digunakan pada penelitian ini adalah 4 dengan *tile* berukuran 8×8 piksel. Pada citra sampel diperoleh sebuah *tile* sebagai berikut (*tile* diperoleh dari Lampiran 5 dengan mengambil potongan 8 baris dan 8 kolom pertama).

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 122 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 122 & 127 \\ 145 & 146 & 0 & 0 & 0 & 0 & 122 & 127 \\ 145 & 146 & 130 & 138 & 0 & 0 & 122 & 127 \end{bmatrix}$$

Selanjutnya, dilakukan *histogram equalization* dengan menggunakan Persamaan (2.20). Untuk intensitas piksel bernilai 0 dengan banyak piksel yaitu $n = 64$ dan banyak nilai intensitas yang mungkin adalah $L = 256$ dapat diperoleh hasil *histogram eaualization* sebagaimana pada Persamaan (4.11).

$$h(0) = round\left(\frac{cdf(0) - cdf_{min}}{64 - cdf_{min}} \times (256 - 1)\right) = 0 \qquad (4.11)$$

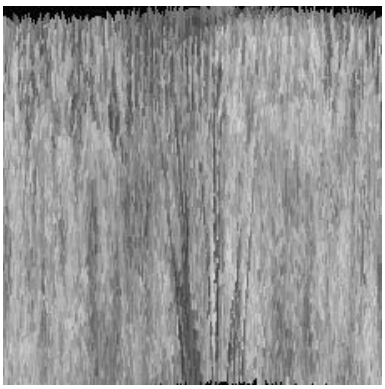
Sedangkan untuk nilai intensitas 122 dapat diperoleh hasil *histogram equalization* seperti pada Persamaan (4.12).

$$h(122) = round\left(\frac{cdf(122) - cdf_{min}}{64 - cdf_{min}} \times (256 - 1)\right) = 78 \qquad (4.12)$$

Perhitungan ini dilakukan terhadap setiap intensitas piksel yang mungkin. Adapun hasil dari *histogram equalization* pada *tile* tersebut dapat ditulis dalam bentuk matriks sebagai berikut.

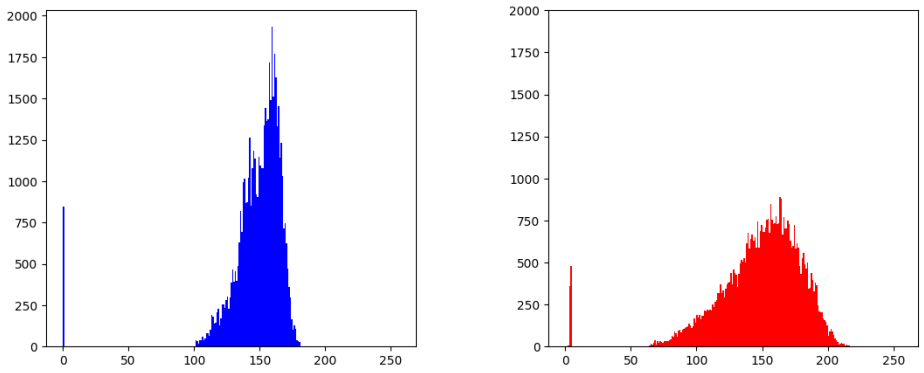
$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 78 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 78 & 137 \\ 216 & 255 & 0 & 0 & 0 & 0 & 78 & 137 \\ 216 & 255 & 157 & 177 & 0 & 0 & 78 & 137 \end{bmatrix}$$

Setelah dilakukan *histogram equalization*, selanjutnya dilakukan *clipping* pada *tile* tersebut. Hal ini dilakukan terhadap seluruh *tile* yang telah dibagi pada citra. Hasil proses CLAHE dapat dilihat pada Gambar 4.11 (matriks hasil *image enhancement* dapat dilihat pada Lampiran 6).



Gambar 4.11 Citra Hasil *Image Enhancement*

Adapun perbedaan histogram antara histogram citra awal dengan citra hasil *image enhancement* dapat dilihat pada Gambar 4.12.



Gambar 4.12 Perbandingan Histogram Setelah *Image Enhancement* (a) Histogram sebelum *image enhancement* (b) Histogram setelah *image enhancement*

4.4 Ekstraksi Fitur

Pada tahap ekstraksi fitur, digunakan algoritma *transfer learning* model AlexNet. Dengan demikian, citra hasil *pre-processing* akan digunakan sebagai data *input* pada model ini. Berdasarkan Tabel 2.1, model AlexNet menerima data *input* dengan dimensi $227 \times 227 \times 3$ yang berarti citra *input* harus memiliki tinggi sebesar 227, lebar sebesar 227 dan kanal sebanyak 3. Perhatikan bahwa citra yang dihasilkan dari proses *pre-processing* sebelumnya merupakan citra *grayscale* dimana hanya memiliki 1 kanal yang mewakili intensitas cahaya saja. Oleh karena itu, untuk mendapatkan data *input* yang valid, maka setiap piksel pada citra hasil *pre-processing* akan dimodifikasi sehingga memiliki 3 kanal dimana nilai intensitas setiap kanal akan memiliki nilai yang sama berdasarkan nilai kanal relatifnya. Berdasarkan pada data sampel, maka citra yang akan digunakan sebagai *input* pada model AlexNet adalah sebagai berikut (selengkapnya dapat dilihat pada Lampiran 6).

$$\begin{bmatrix} 4 & 4 & 4 & \cdots & 3 \\ 4 & 4 & 4 & \cdots & 3 \\ 4 & 4 & 4 & \cdots & 3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 170 & 181 & 174 & \cdots & 180 \end{bmatrix}$$

Tahap ekstraksi fitur menggunakan model AlexNet dilakukan melalui beberapa *layer* yang telah didefinisikan. Secara umum, pada model AlexNet terdapat *layer* konvolusi, *pooling*, *flatten* dan *fully-connected*. Pada model AlexNet, *layer* pertama merupakan *layer* konvolusi dengan ukuran kernel yang digunakan yaitu 11×11. Sebagai contoh, digunakan salah satu jendela berukuran 11×11 dari data sampel sebagai berikut (selengkapnya dapat dilihat pada Lampiran 7).

$$\begin{bmatrix} 4 & 4 & \cdots & 4 \\ 4 & 4 & \cdots & 4 \\ 4 & 4 & \cdots & 4 \\ \vdots & \vdots & \ddots & \vdots \\ 152 & 155 & \cdots & 106 \end{bmatrix}$$

4.4.1 Konvolusi

Pada proses konvolusi *layer* 1, digunakan kernel filter yang juga berukuran 11×11 dengan matriks kanal pertama sebagai berikut (selengkapnya dapat dilihat pada Lampiran 8).

$$\begin{bmatrix} 0{,}0019 & \cdots & 0{,}0025 \\ 0{,}0037 & \cdots & 0{,}0022 \\ 0{,}0047 & \cdots & 0{,}0037 \\ \vdots & \ddots & \vdots \\ 0{,}0033 & \cdots & 0{,}0018 \end{bmatrix}$$

Selanjutnya setiap matriks dan dimensi yang bersesuaian akan dikalikan sehingga diperoleh matriks sebagai berikut (selengkapnya dapat dilihat pada Lampiran 9).

$$\begin{bmatrix} 0{,}0075 & \cdots & 0{,}0101 \\ 0{,}0149 & \cdots & 0{,}0088 \\ 0{,}0189 & \cdots & 0{,}0147 \\ \vdots & \ddots & \vdots \\ 0{,}4976 & \cdots & 0{,}1901 \end{bmatrix}$$

Kemudian seluruh bilangan pada matriks tersebut akan dijumlahkan sehingga dihasilkan nilai intensitas baru yaitu 40,8721. Operasi ini dilakukan berulang terhadap seluruh jendela berukuran 11×11 dari citra sampel sehingga diperoleh hasil dengan ukuran 55×55. Berdasarkan model AlexNet, hasil dari proses konvolusi pertama merupakan matriks berukuran $55 \times 55 \times 96$. Untuk itu, proses konvolusi ini akan dilakukan sebanyak 96 kali dengan menggunakan 96 kernel filter yang berbeda. Dengan demikian, hasil dari *layer* pertama dapat ditulis sebagai matriks berikut (selengkapnya dapat dilihat pada Lampiran 10).

$$\begin{bmatrix} 40{,}8721 & \cdots & 0{,}0000 \\ 19{,}7277 & \cdots & 33{,}0697 \\ 36{,}4977 & \cdots & 34{,}6683 \\ \vdots & \ddots & \vdots \\ 4{,}7239 & \cdots & 48{,}7047 \end{bmatrix}$$

4.4.2 *Pooling*

Setelah proses konvolusi pada *layer* pertama, selanjutnya dilakukan proses *pooling* pada *layer* kedua. Pada model AlexNet, metode *pooling* yang digunakan adalah metode *max pooling* dengan ukuran kernel 3×3. Pada proses *pooling* setiap jendela berukuran 3×3 pada citra hasil *layer* sebelumnya akan disatukan menggunakan nilai piksel yang memiliki nilai maksimum. Dengan demikian, pada *layer* kedua dihasilkan matriks berukuran $27 \times 27 \times 96$ sebagai berikut (selengkapnya dapat dilihat pada Lampiran 11).

$$\begin{bmatrix} 48{,}1110 & \cdots & 46{,}3061 \\ 48{,}1110 & \cdots & 55{,}8968 \\ 50{,}1785 & \cdots & 57{,}6107 \\ \vdots & \ddots & \vdots \\ 53{,}8442 & \cdots & 48{,}8480 \end{bmatrix}$$

Selanjutnya, hasil dari proses *pooling* pada *layer* kedua akan diproses pada *layer* ketiga dan seterusnya dengan cara yang sama baik konvolusi maupun *pooling* sehingga tiba pada *layer fully connected* pertama.

4.4.3 *Fully Connected*

Pada *layer fully connected* pertama, matriks hasil dari *layer-layer* akan didatarkan menjadi vektor terlebih dahulu melalui proses *flatten* sehingga dihasilkan vektor berukuran 9216×1 . Kemudian, vektor tersebut akan disubstitusikan dalam 4096 model matematika. Setiap model tersebut memiliki nilai bobot (*weight*) dan bias yang telah ditentukan. Dengan demikian, hasil akhir dari *layer fully connected* adalah vektor berukuran 4096×1.

Pada penelitian ini, model AlexNet hanya akan diambil hingga *layer fully connected* pertama saja. Dengan demikian, hasil akhir dari proses ekstraksi fitur merupakan vektor berukuran 4096×1 sebagai berikut (selengkapnya dapat dilihat pada Lampiran 19).

$$[0 \quad 11{,}5887 \quad 10{,}385 \quad 0 \quad 19{,}4808 \quad 3{,}8532 \quad \cdots \quad 0]^{-1}$$

4.5 Klasifikasi Citra

Terdapat dua tahap utama yang perlu dilakukan dalam proses klasifikasi citra, yaitu tahap latihan (*training*) dan tahap pengujian (*testing*). Tahap *training* dilakukan untuk menghasilkan model *machine learning* dengan menggunakan data

latih yang telah disediakan, sedangkan tahap *testing* dilakukan untuk menguji dan mengevaluasi model *machine learning* yang telah dilatih sebelumnya. Pada penelitian ini, 630 sampel data citra digunakan sebagai data latih dan 126 data lainnya digunakan sebagai data uji.

Pada tahap *training*, 630 data latih digunakan untuk menghasilkan model *random forest*. Pada dasarnya, model *random forest* merupakan model yang dapat melakukan klasifikasi dengan menggunakan rata-rata prediksi dari beberapa model *decision tree*. Pada penelitian ini, dibuat 300 model *decision tree* untuk membangun model *random forest*.

Tabel 4.1 Fitur Data Latih

No.	Fitur A	Label
1	0,01925	1
2	0,0202	1
3	0,0204	2
4	0,0204	2
5	0,0238	3
6	0,0205	3
7	0,0219	4
8	0,0212	4
9	0,0185	5
10	0,0173	5
⋮	⋮	⋮
630	0,0244	108

Pembuatan decision tree dilakukan dengan menentukan satu fitur yang akan digunakan sebagai root node. Untuk menentukan root node, akan dihitung nilai information gain dari setiap fitur pada data menggunakan Persamaan (2.2). Sebagai contoh dalam menghitung nilai information gain, diberikan salah satu fitur (fitur A)

dari setiap citra beserta labelnya pada Tabel 4.1 (selengkapnya dapat dilihat pada Lampiran 21).

Pada penelitian ini, diambil 5 data pada setiap kelas dari seluruh data sampel. Dengan demikian, dapat diperoleh nilai *entropy* dari seluruh atribut dalam fitur tersebut dengan menggunakan Persamaan (2.1) sebagaimana tertulis pada Persamaan (4.13).

$$\begin{aligned}\text{Entropy}(S) &= \sum_{i=1}^{c} -p_i \log_2 p_i \\ &= -\frac{5}{630}\log_2\frac{5}{630} + \cdots + \left(-\frac{6}{630}\log_2\frac{6}{630}\right) = \\ &6{,}7518\end{aligned} \tag{4.13}$$

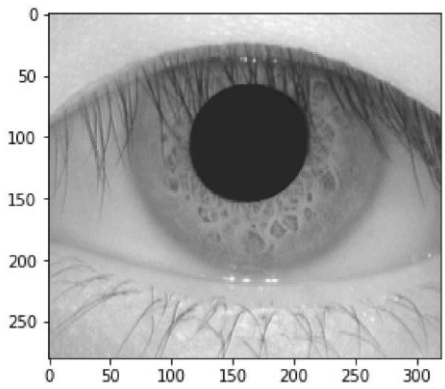
Berdasarkan Tabel 4.1, dapat diperoleh nilai *entropy* dari setiap atribut yang mungkin dari fitur yang ada. Sebagai contoh, dihitung nilai *entropy* untuk setiap atribut dalam fitur yang memiliki nilai kurang dari $0{,}02$. Dengan demikian, diperoleh nilai *entropy* atribut pada Persamaan (4.14).

$$\begin{aligned}\text{Entropy}\left(S_{v<0{,}02}\right) &= -\frac{1}{95}\log_2\frac{1}{95} + \cdots + \left(-\frac{3}{95}\log_2\frac{3}{95}\right) + \\ &\quad \left(-\frac{0}{95}\log_2\frac{0}{95}\right) \\ &= 5{,}3117\end{aligned} \tag{4.14}$$

Selanjutnya hal ini akan dilakukan terhadap seluruh atribut yang mungkin dalam satu fitur tersebut. Setelah diperoleh nilai *entropy* dari setiap atribut maka dapat diperoleh nilai *information gain* dari fitur tersebut dengan menggunakan Persamaan (2.2) sehingga diperoleh hasil seperti pada Persamaan (4.15).

$$\begin{aligned}\text{Gain}(S, A) &= \text{Entropy}(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|}\,\text{Entropy}\,(S_v) \\ &= 6{,}7518 - \left(\frac{95}{630}\,5{,}3117 + \cdots\right) = 4{,}3062\end{aligned} \tag{4.15}$$

Perhitungan *information gain* selanjutnya dilakukan terhadap seluruh fitur yang ada pada data latih. Pada penelitian ini, banyak fitur setiap data sampel adalah 4096 fitur. Fitur yang memiliki nilai *information gain* tertinggi akan dipilih menjadi *root node* pada *decision tree*. Selanjutnya, perhitungan yang sama akan diulangi untuk menentukan cabang (*branches*) pertama dari *root node* yang telah dipilih diantara fitur yang tersisa. Hal ini dilakukan secara berulang sehingga terbentuk seluruh *branches* hingga *leaf nodes* dari *decision tree* tersebut. Dengan demikian, dapat diperoleh *decision tree* pertama yang dapat digunakan untuk memprediksi data.



Gambar 4.13 Sampel Data Uji

Setelah diperoleh model *decision tree* pertama, selanjutnya langkah ini akan diulang sebanyak 300 kali untuk menghasilkan 300 model *decision tree*. Pada setiap iterasi, data latih akan dimodifikasi terlebih dahulu untuk menghasilkan model *decision tree* yang berbeda. Metode modifikasi data tersebut dapat dilakukan dengan menghapus atau menduplikasi beberapa data yang ada. Seluruh model *decision tree* yang telah dibuat selanjutnya akan digunakan untuk membangun model *random forest*.

Tahap terakhir dalam proses klasifikasi adalah tahap *testing*. Dalam tahap ini, model *random forest* yang telah dibuat akan digunakan untuk klasifikasi setiap data uji. Pada *random forest* setiap data akan diprediksi menggunakan seluruh *decision tree* yang terhimpun dalam *random forest*. Proses ini akan menghasilkan 300 hasil klasifikasi dari 300 *decision tree*.

Adapun hasil proses klasifikasi dari salah satu data uji pada Gambar 4.13 dapat dilihat pada Tabel 4.2 (selengkapnya dapat dilihat pada Lampiran 22).

Tabel 4.2 Hasil Klasifikasi *Random Forest*

Model	**Prediksi**
Decision Tree 1	4
Decision Tree 2	2
Decision Tree 3	4
Decision Tree 4	4
Decision Tree 5	9
Decision Tree 6	3
Decision Tree 7	9
Decision Tree 8	4
Decision Tree 9	4
⋮	⋮
Decision Tree 300	4

Hasil klasifikasi *random forest* dapat dicari dengan menghitung rata-rata nilai prediksi dari seluruh *decision tree*. Berdasarkan Tabel 4.2 maka diperoleh hasil klasifikasi salah satu data uji yaitu bernilai 4.

4.6 Evaluasi

Tahap evaluasi pada penelitian ini dilakukan dengan menggunakan teknik k-*fold cross validation*. Nilai *k* yang digunakan pada penelitian ini adalah 6.

Dengan demikian, data sampel berjumlah 756 citra akan dibagi menjadi 6 bagian yang sama sehingga masing-masing bagian memiliki 126 data.

Setelah data dibagi menjadi 6 bagian yang sama, selanjutnya proses klasifikasi baik *training* maupun *testing* dilakukan sebanyak 6 percobaan. Pada setiap percobaan, dihitung nilai akurasi sebagai bahan evaluasi. Sebagai contoh, digunakan hasil klasifikasi pada suatu percobaan yang menghasilkan 108 prediksi bernilai benar dari 126 data uji. Nilai akurasi dari percobaan tersebut dapat dihitung menggunakan Persamaan (2.3) sehingga diperoleh hasil sebagaimana pada Persamaan (4.16).

$$Akurasi_1 = \frac{108}{126} = 0{,}8571 \tag{4.16}$$

Adapun nilai akurasi dari masing-masing percobaan dapat dilihat pada Tabel 4.3. Nilai akurasi akhir dari penelitian ini adalah rata-rata nilai akurasi dari masing-masing percobaan dapat ditulis sebagaimana pada Persamaan (4.17).

$$\begin{aligned} Akurasi &= \frac{85{,}71 + 86{,}50 + 85{,}71 + 85{,}71 + 86{,}50 + 85{,}71}{6} \\ &= 85{,}98 \end{aligned} \tag{4.17}$$

Dengan demikian, diperoleh hasil akhir akurasi dari penelitian ini ialah sebesar 85,98%.

Tabel 4.3 Hasil K-*fold Cross Validation*

No. Percobaan	**Akurasi**
Percobaan 1	85,71%
Percobaan 2	86,50%
Percobaan 3	85,71%
Percobaan 4	85,71%
Percobaan 5	86,50%
Percobaan 6	85,71%

4.7 Implementasi *Transfer Learning* dan Algoritma *Random Forest* pada Identifikasi Iris Mata dengan Nilai Agama

Pada penelitian ini, diperoleh bahwa implementasi *transfer learning* dan algoritma *random forest* pada identifikasi iris mata menghasilkan tingkat akurasi sebesar 85,98%. Dengan demikian, model yang diperoleh dari implementasi pada penelitian ini cukup memenuhi untuk digunakan dalam sistem keamanan data maupun sebagai sistem identifikasi personal. Selain itu, hal ini juga menunjukkan bahwa iris mata manusia memang benar memiliki karakteristik yang berbeda sehingga dapat diidentifikasi menggunakan algoritma *machine learning* yang ada.

Dalam agama Islam, disebutkan bahwa manusia diciptakan oleh Allah SWT dalam bentuk yang sebaik-baiknya. Hal ini tertulis dalam Al-Qur'an surat At-Tin ayat 4. Berdasarkan ayat tersebut, diperoleh bahwa setiap hal yang terdapat pada tubuh manusia baik secara jiwa dan raga diciptakan oleh Allah SWT dengan sebaik mungkin. Salah bukti akan keistimewaan penciptaan manusia dapat dilihat pada iris mata manusia.

Berdasarkan penelitian, diketahui bahwa setiap manusia memiliki karakteristik iris mata yang berbeda. Hal ini dapat terlihat bahwasanya setiap iris mata dari individu yang berbeda dalam penelitian ini memiliki vektor fitur yang berbeda, sedangkan iris mata dari individu yang sama memiliki vektor fitur yang cenderung sama. Dengan demikian, dari proses implementasi *transfer learning* dan algoritma *random forest* pada identifikasi iris mata dalam penelitian ini juga dapat menunjukkan bahwasanya setiap iris mata manusia memiliki ciri dan karakteristik serta vektor fitur yang berbeda. Hal ini dapat menyimpulkan bahwasanya Allah SWT menciptakan iris mata setiap manusia dengan penuh perhitungan. Hal ini

sejalan dengan firman Allah SWT dalam Al-Qur’an surat Al-Qamar ayat 49. Berdasarkan ayat tersebut, diketahui bahwa segala ciptaan Allah SWT baik yang ada di langit maupun di bumi diciptakan dengan ukuran dan perhitungannya masing-masing, demikian pula dengan iris mata pada manusia. Dengan demikian, secara tidak langsung penelitian ini juga menunjukkan akan bukti kebesaran Allah SWT melalui ciptaannya yang penuh dengan perhitungan, yaitu iris mata. Oleh karena itu, dengan adanya penelitian ini diharapkan penulis maupun pembaca dapat semakin meningkatkan keimanan diri kepada Allah SWT.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan penjelasan sebelumnya, dihasilkan kesimpulan sebagai berikut.

1. Penelitian ini dilakukan melalui tiga tahap utama yaitu *preprocessing*, ekstraksi fitur, dan klasifikasi. Tahap *preprocessing* pada penelitian ini terbagi menjadi tahap lokalisasi iris menggunakan algoritma *integro-differensial*, segmentasi iris dengan metode *masking*, normalisasi iris dengan metode *rubber sheet* dan *image enhancement* menggunakan metode CLAHE. Tahap ekstraksi fitur dilakukan menggunakan metode *transfer learning* dengan model AlexNet sehingga menghasilkan vektor fitur berdimensi 4096×1. Tahap klasifikasi menggunakan algoritma *random forest* dengan jumlah *decision tree* yaitu 300.
2. Proses evaluasi dalam penelitian ini dilakukan dengan algoritma k-*fold cross validation* dengan nilai $k = 6$. Hasil akhir dari evaluasi tersebut diperoleh tingkat akurasi $85{,}98\%$ dengan 6 kali percobaan terhadap 756 data citra iris mata.

5.2 Saran

Berdasarkan latar belakang pada penelitian ini yaitu terkait kegunaan *machine learning* pada identifikasi iris mata dalam sistem keamanan data, maka

terdapat beberapa hal yang dapat disarankan oleh peneliti untuk mengembangkan penelitian ini sebagai berikut.

1. Implementasi algoritma dalam bentuk aplikasi baik berupa *hardware* ataupun *software*. Hal ini merupakan salah satu bentuk upaya dalam mewujudkan sistem keamanan yang lebih baik dari masa ke masa.
2. Melakukan penelitian menggunakan algoritma lainnya terutama dalam tahap ekstraksi fitur atau tahap klasifikasi dengan harapan dapat memperoleh tingkat akurasi yang lebih baik. Mengingat salah satu kegunaan algoritma ini sebagai sistem keamanan, maka akan lebih baik jika tingkat akurasi dapat ditingkatkan hingga melebihi 90%.

DAFTAR PUSTAKA

Alaoui, S.S., Farhaoui, Y. & Aksasse, B. (2018). Classification algorithms in Data Mining. International Journal of Tomography and Simulation. 31. 34-44

Ali, Amjad, dkk. (2023). Demystifying CNN with Mathematical Insights: A Prelude with Application to an AI-based Sustainable Solution for Diabetic Retinopathy Diagnosis. doi:10.21203/rs.3.rs-3338196/v1

Amazon Web Services. n.d. Apa itu Deep Learning? - Penjelasan tentang Deep Learning - AWS. Diambil dari https://aws.amazon.com/id/what-is/deep-learning/

Andono, N.P., Sutojo, T. dan Muljono. (2017). Pengolahan Citra Digital. Yogyakarta: Andi

Arvianna, Geofanni Nerissa. (2022). Decision Tree: Pengertian, Plus Minus, dan Cara Membuatnya. Diambil dari https://glints.com/id/lowongan/decision-tree-adalah/#.ZCkosnZBy00

Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5–32. doi:10.1023/a:1010933404324

Daugman, J. (2004). How Iris Recognition Works. *IEEE Transactions on Circuits and Systems for Video Technology*, *14*(1), 21–30. https://doi.org/10.1109/TCSVT.2003.818350

Dharmaraj. (2022). Image classification and prediction using transfer learning. Diambil dari https://medium.com/@draj0718/image-classification-and-prediction-using-transfer-learning-3cf2c736589d

Eddine, Guissous Alla. (2019). Skin Lesion Classification Using Deep Neural Network. doi:10.48550/arXiv.1911.07817

FSCJ. n.d. The Human Eye - Astronomy Lab. Diambil dari https://fscj.pressbooks.pub/astronomy/chapter/the-human-eye/

G. Alaslani, M., & A. Elrefaei, L. (2018). Convolutional Neural Network Based Feature Extraction for IRIS Recognition. International Journal of Computer Science and Information Technology, 10(2), 65–78. https://doi.org/10.5121/ijcsit.2018.10206

Hakim, Ahmad Zidan Nur. (2022). Implementasi metode FIRE dan Image Processing Citra untuk Mendeteksi Pola Iris pada Proses Autentikasi Smartphone. Undergraduate thesis, Universitas Islam Negeri Maulana Malik Ibrahim

Hakim, R.B. Fajriya. (2019). Decision Tree. Diambil dari

https://medium.com/@986110101/decision-tree-d7ed1705be7

Hu, Q., Yin, S., Ni, H., & Huang, Y. (2020). An End to End Deep Neural Network for Iris Recognition. In Procedia Computer Science (Vol. 174, pp. 505–517). Elsevier B.V. https://doi.org/10.1016/j.procs.2020.06.118

Ipe, Vineetha & Thomas Kallivayalil, Tony. (2020). CNN Based Periocular Recognition Using Multispectral Images. doi:10.1007/978-981-15-4828-4_9

Jauhari, Raihan Nugroho. (2020). Computer Vision menggunakan Transfer Learning di Keras. Diambil dari https://raihanrnj.medium.com/computer-vision-menggunakan-transfer-learning-di-keras-6015c51c016c

Kaggle. n.d. L24 Convolutional neural network. Diambil dari https://www.kaggle.com/code/jhskaggle/l24-convolutional-neural-networks

Kemenag RI. (2022). Qur'an Kemenag. Lajnah Pentashihan Mushaf Al-Qur'an. Diambil dari https://quran.kemenag.go.id/

Krizhevsky, A., Sutskever, I. dan Hinton, G.E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Proceedings of the 25th International Conference on Neural Information Processing Systems, Volume 1, Lake Tahoe, 3-6 Desember 2012, 1097-1105

Kusuma, Purba Daru. (2020). Machine Learning Teori, Program, dan Studi Kasus. Yogyakarta: Deepublish

Llorella, Fabio (2022). Classify four imagined objects with EEG signals. Evolutionary Intelligence. 15. doi:10.1007/s12065-021-00577-y

Murphy, Kevin P. (2012). Machine Learning: A Probabilistic Perspective. London: The MIT Press

Pandian, Shanthababu. (2023). K-Fold Cross Validation Technique and its Essentials. Diambil dari https://www.analyticsvidhya.com/blog/2022/02/k-fold-cross-validation-technique-and-its-essentials/#h-what-is-accuracy-of-the-model-and-performance

Patel, Dhiren R. (2008). Information Security Theory and Practice. India: PHI Learning

Petrou, M., & Petrou, C. (2010). Image Processing: The Fundamentals (pp. 1–794). Wiley. https://doi.org/10.1002/9781119994398

Pizer, S. M., Amburn, E. P., Austin, J. D., dkk. (1987). Adaptive histogram equalization and its variations. Computer Vision, Graphics, and Image Processing, 39(3), 355–368. doi:10.1016/s0734-189x (87) 80186-x

Rahman, Sayuti, dkk. (2021). Convolutional Neural Networks untuk Visi Komputer Jaringan Saraf. Yogyakarta: Deepublish

Rezika, A., Ernawati, E., & Erlansari, A. (2018). Identifikasi Pola Iris Mata Dengan Algoritme Daugman dan Metode Hamming Distance. Rekursif: Jurnal Informatika, 6(2). https://doi.org/10.33369/rekursif.v6i2.4277

Setiawan, Wahyudi. (2020). Deep Learning menggunakan Convolutional Neural Network: Teori dan Aplikasi. Malang: MNC

SmartDraw. (2023). Decision Tree - Learn Everything About Decision Tree. Diambil dari https://www.smartdraw.com/decision-tree/

Spotfire. (2023). Demystifying the Random Forest Algorithm for Accurate Predictions. Diambil dari https://www.spotfire.com/glossary/what-is-a-random-forest

Sudhakar, Shreenidhi. (2017). Histogram Equalization. Diambil dari https://towardsdatascience.com/histogram-equalization-5d1013626e64

Sumijan, Purnama, P.A.W., & Arlis, S. (2021). Teknologi Biometrik: Implementasi pada Bidang Medis Menggunakan Matlabs. Sumatera Barat: Insan Cendekia Mandiri

Tran, Hieu. (2019). Survey of Machine Learning and Data Mining Techniques used in Multimedia System. doi:10.13140/RG.2.2.20395.49446/1

Vinay, A., dkk. (2020). Enhancement of Degraded CCTV Footage for Forensic Analysis. International Conference on Innovative Computing and Communications, 617–636. doi:10.1007/978-981-15-5113-0_50

Yoss, Andrea. (2020). Transfer Learning using Pre-Trained AlexNet Model and Fashion-MNIST. Diambil dari https://towardsdatascience.com/transfer-learning-using-pre-trained-alexnet-model-and-fashion-mnist-43898c2966fb

LAMPIRAN

Lampiran 1 Matriks Citra Sampel

https://bit.ly/cropped_009_1_1

Lampiran 2 Lingkar Calon Pupil 1
https://bit.ly/lingkar_calon_pupil_1

Lampiran 3 Hasil *Integro-differensial*
https://bit.ly/hasil_integro_differensial

Lampiran 4 Hasil Segmentasi Iris

https://bit.ly/segmented_009_1_1

Lampiran 5 Hasil Normalisasi Iris

https://bit.ly/normalized_009_1_1

Lampiran 6 Hasil *Image Enhancement*

https://bit.ly/clahe_009_1_1

Lampiran 7 *Window* Pertama Dari Citra Hasil *Image Enhancement*

https://bit.ly/clahe_window1_009_1_1

Lampiran 8 Kernel Filter Pertama Tahap Konvolusi Pertama

https://bit.ly/kernel_filter_11x11

Lampiran 9 Hasil Filter Pertama Pada Tahap Konvolusi

https://bit.ly/clahe_window1_filtered_11x11_009_1_1

Lampiran 10 Hasil Tahap Konvolusi 1

https://bit.ly/konvolusi1_009_1_1

Lampiran 11 Hasil Tahap *Pooling* 1
https://bit.ly/pooling1_009_1_1

Lampiran 12 Hasil Tahap Konvolusi 2
https://bit.ly/konvolusi2_009_1_1

Lampiran 13 Hasil Tahap *Pooling* 2
https://bit.ly/pooling2_009_1_1

Lampiran 14 Hasil Tahap Konvolusi 3
https://bit.ly/konvolusi3_009_1_1

Lampiran 15 Hasil Tahap Konvolusi 4
https://bit.ly/konvolusi4_009_1_1

Lampiran 16 Hasil Tahap Konvolusi 5
https://bit.ly/konvolusi5_009_1_1

Lampiran 17 Hasil Tahap *Pooling* 3
https://bit.ly/pooling5_009_1_1

Lampiran 18 Hasil Tahap *Flatten*
https://bit.ly/flatten_009_1_1

Lampiran 19 Hasil Tahap Fully Connected Pertama
https://bit.ly/fully_connected_6_009_1_1

Lampiran 20 Fitur A Dari Seluruh Data
https://bit.ly/feature_A_full

Lampiran 21 Fitur A Dari Data Latih
https://bit.ly/feature_A_train

Lampiran 22 Hasil Klasifikasi Terhadap Salah Satu Data Uji

https://bit.ly/hasil_klasifikasi_1

Lampiran 23 Kode Python

Impor *Library*

```
import cv2 as cv
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import os
import glob
import fnmatch
from tqdm import tqdm
from google.colab.patches import cv2_imshow
import itertools
import math
from typing import Tuple, List

import seaborn as sns

from keras.models import Model, Sequential
from keras.layers import Dense, Dropout, Flatten, Conv2D,
MaxPooling2D, BatchNormalization
from keras.callbacks import EarlyStopping
from keras.utils import to_categorical

from sklearn import svm
from sklearn import preprocessing
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix,
precision_score, recall_score, ConfusionMatrixDisplay
from sklearn.model_selection import RandomizedSearchCV,
train_test_split
from scipy.stats import randint

main = '/content/drive/MyDrive/CASIA-IrisV1'
```

Fungsi Operator *Integro-Differensial*

```
# Resource : https://github.com/banderlog/daugman

def daugman(gray_img: np.ndarray, center: Tuple[int, int],
```

```
            start_r: int, end_r: int, step: int = 1) ->
Tuple[float, int]:
    """ The function will calculate pixel intensities for the
circles
        in the ``range(start_r, end_r, step)`` for a given
``center``,
        and find a circle that precedes the biggest intensity drop

        :param gray_img: grayscale picture
        :param center:  center coordinates ``(x, y)``
        :param start_r: bottom value for iris radius in pixels
        :param end_r: top value for iris radius in pixels
        :param step: step value for iris radii range in pixels

        .. attention::
            Input grayscale image should be a square, not a
rectangle

        :return: intensity_value, radius
    """
    x, y = center
    intensities = []
    mask = np.zeros_like(gray_img)

    # for every radius in range
    radii = list(range(start_r, end_r, step))  # type: List[int]
    for r in radii:
        # draw circle on mask
        cv.circle(mask, center, r, 255, 1)
        # get pixel from original image, it is faster than np or
cv2
        diff = gray_img & mask
        # normalize, np.add.reduce faster than .sum()
        #            diff[diff > 0] faster than .flatten()
        intensities.append(np.add.reduce(diff[diff > 0]) / (2 *
math.pi * r))
        # refresh mask
        mask.fill(0)

    # calculate delta of radius intensitiveness
    #     mypy does not tolerate var type reload
    intensities_np = np.array(intensities, dtype=np.float32)
    del intensities

    # circles intensity differences, x5 faster than np.diff()
    intensities_np = intensities_np[:-1] - intensities_np[1:]
    # aply gaussian filter
```

```
    #    GaussianBlur() faster than filter2D() with custom kernel
    # original kernel:
    # > The Gaussian filter in our case is designedin MATLAB and
    # > is a 1 by 5 (rows by columns) vector with intensity values
    # > given by vector A = [0.0003 0.1065 0.7866 0.1065 0.0003]
    intensities_np = abs(cv.GaussianBlur(intensities_np, (1, 5),
0))
    # get maximum value
    idx = np.argmax(intensities_np)  # type: int

    # return intensity value, radius
    return intensities_np[idx], radii[idx]

def find_iris(gray: np.ndarray, *,
              daugman_start: int, daugman_end: int,
              daugman_step: int = 1, points_step: int = 1,) ->
Tuple[Tuple[int, int], int]:
    """ The function will apply :func:`daugman` on every pixel in
the calculated image slice.
        Basically, we are calculating where lies set of valid
circle centers.
        It is assumed that iris center lies within central 1/3 of
the image.

        :param gray: graysacale **square** image
        :param points_step:  it  will  run  daugman  for  each
``points_step``th point.
                             It  has  linear  correlation  with
overall iris search speed
        :param daugman_start: bottom value for iris radius in
pixels for :func:``daugman``
        :param daugman_end: top value for iris radius in pixels
for :func:``daugman``
        :param daugman_step: step value for iris radii range in
pixels for :func:``daugman``.
                              It  has  linear  correlation  with
overall iris search speed

        :return: radius with biggest intensiveness delta on image
as ``((xc, yc), radius)``
    """
    h, w = gray.shape
    if h != w:
        print('Your image is not a square!')

    # reduce step for better accuracy
    # we will look only on dots within central 1/3 of image
```

```
    single_axis_range = range(int(h / 3), h - int(h / 3),
points_step)
    all_points      =      itertools.product(single_axis_range,
single_axis_range)

    intensity_values = []
    coords = []  # List[Tuple[Tuple(int, int), int]]

    for point in all_points:
        val, r = daugman(gray, point, daugman_start, daugman_end,
daugman_step)
        intensity_values.append(val)
        coords.append((point, r))

    # return the radius with biggest intensiveness delta on image
    # ((xc, yc), radius)
    # x10 faster than coords[np.argmax(values)]
    best_idx = intensity_values.index(max(intensity_values))
    return coords[best_idx]
```

Fungsi Lokalisasi Iris

```
def iris_localization(src):
  gray = cv.cvtColor(src,cv.COLOR_BGR2GRAY)

  pupil   =   find_iris(gray,daugman_start=30,   daugman_end=60,
daugman_step=1, points_step=2)
  center, r_pupil = pupil

  # _, r_iris = daugman(gray, center, 60, 120, 1)
  # _, r_iris

  r_iris = r_pupil + 40
  rad = (r_pupil,r_iris)

  result = src.copy()
  cv.circle(result, center, r_pupil, (255,255,255), 1)
  cv.circle(result, center, r_iris, (255,255,255), 1)

  # print("Iris and Pupil Center : " + str(center))
  # print("Iris Radian : " + str(r_iris))
  # print("Pupil Radian : " + str(r_pupil))

  return center, rad, result
```

Fungsi Segmentasi Iris

```
def iris_segmentation(src, center, r_pupil, r_iris):
  gray = cv.cvtColor(src, cv.COLOR_BGR2GRAY)
  mask = np.zeros(gray.shape, np.uint8)
  cv.circle(mask, center, r_iris, (255, 255, 255), cv.FILLED, 1)
  cv.circle(mask, center, r_pupil, (0, 0, 0), cv.FILLED, 1)
  result = cv.bitwise_and(gray, gray, mask=mask)

  return result, mask
```

Fungsi Memotong Citra

```
def iris_denoising(gray, mask):
  contours,     _     =     cv.findContours(mask,     cv.RETR_TREE,
cv.CHAIN_APPROX_SIMPLE)
  _, binary = cv.threshold(gray, 100, 255, cv.THRESH_BINARY_INV)
  dst = cv.inpaint(gray,binary,5,cv.INPAINT_TELEA)
  _, binary2 = cv.threshold(dst, 180,255, cv.THRESH_BINARY)
  dst = cv.inpaint(dst,binary2,10,cv.INPAINT_TELEA)
  masked = cv.bitwise_and(dst, dst, mask=mask)

  selected = max(contours,key=cv.contourArea)
  x,y,w,h = cv.boundingRect(selected)
  cropped = masked[y:y+h,x:x+h]

  return cropped
```

Fungsi Normalisasi Iris

```
# Resource : https://github.com/YifengChen94/IrisReco

def iris_normalization(image, height, width, r_in, r_out):      #
Daugman 归一化，输入为 640*480,输出为 width*height
    thetas = np.arange(0, 2 * np.pi, 2 * np.pi / width)  # Theta
values

    # Create empty flatten image
    flat = np.zeros((height,width, 3), np.uint8)
    circle_x = int(image.shape[0] / 2)
    circle_y = int(image.shape[1] / 2)

    for i in range(width):
        theta = thetas[i]  # value of theta coordinate
        # get coordinate of boundaries
        Xi = circle_x + r_in * np.cos(theta)
        Yi = circle_y + r_in * np.sin(theta)
```

```
        Xo = circle_x + r_out * np.cos(theta)
        Yo = circle_y + r_out * np.sin(theta)

        for j in range(height):
            r_pro   =   j   /   height      #   value   of   r
coordinate(normalized)

            # the matched cartesian coordinates for the polar
coordinates
            Xc = (1 - r_pro) * Xi + r_pro * Xo
            Yc = (1 - r_pro) * Yi + r_pro * Yo

            color = image[int(Xc)][int(Yc)]  # color of the pixel

            flat[j][i] = color
    return flat  # liang
```

Impor Dan *Pre-processing* Data

```
# Capture training, testing data and their labels into respective
lists
train_images = []
test_images = []
train_iris = []
test_iris = []
train_labels = []
test_labels = []

print("Importing dataset...\n")

for dir in tqdm(os.listdir(main)):
  dir_path = os.path.join(main,dir)
  for subdir in os.listdir(dir_path):
    subdir_path = os.path.join(dir_path,subdir)
    test = True
    for    img_path    in    glob.glob(os.path.join(subdir_path,
"*.bmp")):
      img = cv.imread(img_path)

      if test:
        test_images.append(img)
        test_labels.append(int(dir))
        test = False
      else:
        train_images.append(img)
        train_labels.append(int(dir))
```

```
print("------------------------------------------------------------
--------")
print("Preprocessing training images...\n")

for img in tqdm(train_images):
  # Crop the image into square
  h, w = img.shape[:2]
  src = img[:, 30:w-10]
  gray = cv.cvtColor(src, cv.COLOR_BGR2GRAY)

  # Preprocessing the image
  center, rad, localized = iris_localization(src)
  segmented,  mask  =  iris_segmentation(src,  center,  rad[0],
rad[1])
  cropped = iris_denoising(gray, mask)
  normalized = cv.cvtColor(iris_normalization(cropped, 227, 227,
rad[0], rad[1]), cv.COLOR_BGR2GRAY)
  clahe = cv.createCLAHE(3).apply(normalized)

  iris = cv.cvtColor(clahe, cv.COLOR_GRAY2BGR)

  train_iris.append(iris)

print("------------------------------------------------------------
--------")
print("Preprocessing testing images...\n")

for img in tqdm(test_images):
  # Crop the image into square
  h, w = img.shape[:2]
  src = img[:, 30:w-10]
  gray = cv.cvtColor(src, cv.COLOR_BGR2GRAY)

  # Preprocessing the image
  center, rad, localized = iris_localization(src)
  segmented,  mask  =  iris_segmentation(src,  center,  rad[0],
rad[1])
  cropped = iris_denoising(gray, mask)
  normalized = cv.cvtColor(iris_normalization(cropped, 227, 227,
rad[0], rad[1]), cv.COLOR_BGR2GRAY)
  clahe = cv.createCLAHE(3).apply(normalized)

  iris = cv.cvtColor(clahe, cv.COLOR_GRAY2BGR)

  test_iris.append(iris)

# Convert lists to arrays
```

```
train_images = np.array(train_images)
test_images = np.array(test_images)
train_iris = np.array(train_iris)
test_iris = np.array(test_iris)
train_labels = np.array(train_labels)
test_labels = np.array(test_labels)

print("\n-----------------------------------------------------------
----------")
print("Preprocessing DONE!")

#Encode labels from text to integers.
le = preprocessing.LabelEncoder()

le.fit(test_labels)
test_labels_encoded = le.transform(test_labels)
le.fit(train_labels)
train_labels_encoded = le.transform(train_labels)

#Split data into test and train datasets (already split but
assigning to meaningful convention)

x_train,    y_train,    x_test,    y_test    =    train_iris,
train_labels_encoded, test_iris, test_labels_encoded

# Normalize pixel values to between 0 and 1
x_train, x_test = x_train / 255.0, x_test / 255.0
```

Model AlexNet Dan Ekstraksi Fitur

```
# Load Alex-Net Model
model = Sequential()

# Feature Extraction Layer
model.add(Conv2D(96,     (11,11),     4,     activation='relu',
input_shape=(227, 227, 3)))
model.add(MaxPooling2D((3,3), 2))
model.add(Conv2D(256,     (5,5),     1,     activation='relu',
padding='same'))
model.add(MaxPooling2D((3,3), 2))
model.add(Conv2D(384,     (3,3),     1,     activation='relu',
padding='same'))
model.add(Conv2D(384,     (3,3),     1,     activation='relu',
padding='same'))
model.add(Conv2D(256,     (3,3),     1,     activation='relu',
padding='same'))
model.add(MaxPooling2D((3,3), 2))
```

```
model.add(Dropout(0.5))

# Flatten Layer
model.add(Flatten())

# Fully Connected Layer
model.add(Dense(4096,activation='relu'))
# model.add(Dropout(0.5))
# model.add(Dense(4096,activation='relu'))
# model.add(Dense(1000,activation='softmax'))

model.summary()

train_feature = model.predict(x_train)
test_feature = model.predict(x_test)
```

Klasifikasi Citra

```
# Generate Random Forest Classifier
RF_model    =    RandomForestClassifier(n_estimators    =    300,
random_state = 42)
# Train the model on training data
RF_model.fit(train_feature, y_train) # For sklearn no one hot
encoding

# Now predict using the trained RF model.
prediction_RF = RF_model.predict(test_feature)

# Inverse le transform to get original label back.
prediction_RF = le.inverse_transform(prediction_RF)

# Print overall accuracy
print ("Accuracy = ", accuracy_score(test_labels, prediction_RF))
```

Evaluasi Dengan K-*fold Cross Validation*

```
from sklearn.model_selection import cross_val_score

# Generate Random Forest Classifier
RF_model    =    RandomForestClassifier(n_estimators    =    300,
random_state = 42)

scores = cross_val_score(RF_model, X, y, cv=6)
final_scores = np.mean(scores)
```

RIWAYAT HIDUP

Akhmad Roziqin, lahir di Pasuruan pada tanggal 14 Juli 2001, tinggal di Desa Ketegan RT. 03 RW. 01, Kecamatan Rejoso, Kabupaten Pasuruan, Jawa Timur. Penulis merupakan anak pertama dari Bapak Heru Suprayogi dan Ibu Winnuryati. Selama masa pendidikan, penulis menempuh pendidikan dasar di SD Negeri Ketegan dari tahun 2007 hingga tahun 2013, dilanjutkan dengan pendidikan menengah pertama di SMP Negeri 1 Winongan hingga tahun 2016, lalu penulis melanjutkan pendidikan jenjang menengah atas di SMA Negeri 1 Gondangwetan hingga tahun 2019. Setelah lulus dari SMA, penulis melanjutkan pendidikan sebagai mahasiswa program studi Matematika di Universitas Islam Negeri Maulana Malik Ibrahim Malang. Selama kuliah, penulis turut berkontribusi aktif dalam berbagai kegiatan baik internal maupun eksternal kampus. Beberapa pengalaman yang pernah dilalui oleh penulis yaitu sebagai pengurus Himpunan Mahasiswa Jurusan (HMJ) “Integral” Matematika selama dua periode, ketua pelaksana Kompetisi Matematika (KOMET) XX Nasional, ketua Ikatan Mahasiswa Pasuruan (IMAPAS), ketua komunitas Serambi Matematika Aktif (SeMatA), serta asisten laboratorium mata kuliah Pemrograman Komputer dalam tiga semester. Penulis memiliki ketertarikan terhadap bidang pemrograman baik dalam bidan *web development* maupun *data science*. Penulis menerima kritik dan saran atas penelitian ini. Kritik dan saran berkaitan penelitian ini dapat dikirim pada email penulis yaitu roziqinakhmad14juli@gmail.com.

KEMENTERIAN AGAMA RI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Gajayana No.50 Dinoyo Malang Telp. / Fax. (0341)558933

BUKTI KONSULTASI SKRIPSI

Nama	: Akhmad Roziqin
NIM	: 19610087
Fakultas / Program Studi	: Sains dan Teknologi / Matematika
Judul Skripsi	: Implementasi Metode *Transfer Learning* dan Algoritma *Random Forest* pada Identifikasi Iris Mata
Pembimbing I	: Hisyam Fahmi, M.Kom.
Pembimbing II	: Juhari, M.Si

No	Tanggal	Hal	Tanda Tangan
1.	3 November 2022	ACC Pengajuan Topik	1.
2.	14 April 2023	Konsultasi Bab I, II, dan III	2.
3.	9 Mei 2023	Revisi Bab I, II, dan III	3.
4.	10 Mei 2023	ACC Bab I, II, dan III	4.
5.	12 Mei 2023	Konsultasi Kajian Agama Bab I dan II	5.
6.	12 Mei 2023	ACC Kajian Agama Bab I dan II	6.
7.	23 Oktober 2023	ACC Revisi Seminar Proposal	7.
8.	23 Oktober 2023	Konsultasi Bab IV dan V	8.
9.	24 Oktober 2023	Revisi Bab IV dan V	9.
10.	26 Oktober 2023	ACC Bab IV dan V	10.
11.	2 November 2023	Konsultasi Kajian Agama Bab IV	11.
12.	3 November 2023	ACC Kajian Agama Bab IV	12.
13.	4 Desember 2023	ACC Revisi Seminar Hasil	13.
14.	27 Desember 2023	ACC Keseluruhan (Pembimbing I)	14.

KEMENTERIAN AGAMA RI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Gajayana No.50 Dinoyo Malang Telp. / Fax. (0341)558933

15.	27 Desember 2023	ACC Keseluruhan (Pembimbing II)	15.

Malang, 27 Desember 2023
Mengetahui,
Ketua Program Studi Matematika

Dr. Elly Susanti, M.Sc
NIP. 19741129 200012 2 005