IDENTIFIKASI TEMPAT PENGAMBILAN FOTO JALAN DI KOTA MALANG MENGGUNAKAN METODE NEURAL NETWORK

SKRIPSI

Oleh:
PRASETYO PUTRA PRATAMA
NIM. 210605110157

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2025

IDENTIFIKASI TEMPAT PENGAMBILAN FOTO JALAN DI KOTA MALANG MENGGUNAKAN METODE NEURAL NETWORK

SKRIPSI

Diajukan kepada:
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh:
PRASETYO PUTRA PRATAMA
NIM. 210605110157

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2025

HALAMAN PERSETUJUAN

IDENTIFIKASI TEMPAT PENGAMBILAN FOTO JALAN DI KOTA MALANG MENGGUNAKAN METODE NEURAL NETWORK

SKRIPSI

Oleh:
PRASETYO PUTRA PUTRAMA
NIM. 210605110157

Telah Diperiksa dan Disetujui untuk Diuji:
Tanggal: 17 April 2025

Pembimbing I,	Pembimbing II,
Dr. Cahyo Crysdian, M.Cs.	Dr. Irwan Budi Santoso, M.Kom
NIP. 19740424 200901 1 008	NIP. 19770103 201101 1 004

Mengetahui,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Ir. Fachrul Kurniawan, M.MT, IPU
NIP. 19771020 200912 1 001

HALAMAN PENGESAHAN

IDENTIFIKASI TEMPAT PENGAMBILAN FOTO JALAN DI KOTA MALANG MENGGUNAKAN METODE NEURAL NETWORK

SKRIPSI

Oleh:
PRASETYO PUTRA PRATAMA
NIM. 210605110157

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Tanggal: 16 Mei 2025

Susunan Dewan Penguji

Ketua Penguji	: Supriyono, M. Kom NIP. 19841010 201903 1 012	( )
Anggota Penguji I	: Roro Inda Melani, M.T, M.Sc NIP. 19780925 200501 2 008	( )
Anggota Penguji II	: Dr. Cahyo Crysdian, M.Cs. NIP. 19740424 200901 1 008	( )
Anggota Penguji III	: Dr. Irwan Budi Santoso, M.Kom NIP. 19770103 201101 1 004	( )

Mengetahui dan Mengesahkan,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Ir. Fachrul Kurniawan, M.MT, IPU
NIP. 19771020 200912 1 001

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama	: Prasetyo Putra Pratama
NIM	: 210605110157
Fakultas / Program Studi	: Sains dan Teknologi / Teknik Informatika
Judul Skripsi	: Identifikasi Tempat Pengambilan Foto Jalan Di Kota Malang Menggunakan Metode Neural Network

Menyatakan dengan sebenarnya bahwa Skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambil alihan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini merupakan hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 17 April 2025
Yang membuat pernyataan,

Prasetyo Putra Pratama
NIM.210605110157

MOTTO

"Berpikirlah, sampai tidak mampu lagi berfikir."

HALAMAN PERSEMBAHAN

Dengan rasa syukur yang mendalam kepada Allah SWT, yang telah memberikan rahmat, kesehatan, dan kekuatan sehingga skripsi ini dapat terselesaikan. Penulis mempersembahkan karya ini kepada orang paling berharga dalam hidup saya, ibu tercinta dan tersayang, Hamidah. Yang telah membimbing saya tanpa keluh kesah hingga sampai pada titik ini.

KATA PENGANTAR

Bismillahirrahmaanirrahiim, Assalamu'alaikum wr. wb.

Segala puji dan syukur yang tak terhingga penulis panjatkan kepada Allah SWT atas rahmat dan petunjuk-Nya. Berkat karunia-Nya, penulis dapat menyelesaikan penyusunan skripsi yang berjudul "Identifikasi Tempat Pengambilan Foto Jalan Di Kota Malang Menggunakan Metode Neural Network". Oleh karena itu, dengan penuh rendah hati, penulis mengucapkan terima kasih kepada:

1. Prof. Dr. H. M. Zainuddin, M.A., selaku rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Prof. Dr. Hj. Sri Hariani, M.Si., selaku dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Dr. Ir. Fachrul Kurniawan, M.MT, selaku Ketua Program Studi Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang, atas motivasi dan arahan yang diberikan selama proses perkuliahan.
4. Dr. Cahyo Crysdian, M.Cs., selaku dosen Pembimbing 1 yang telah membimbing dan mengajarkan saya banyak pelajaran berharga hingga selesai.
5. Dr. Irwan Budi Santoso, M.Kom, selaku dosen pembimbing II, yang dengan penuh kesabaran membimbing penulis hingga selesai
6. Supriyono, M. Kom dan Roro Inda Melani, M.T, M.Sc, selaku dosen penguji saya, yang telah memberikan kritik dan saran yang berharga selama proses ujian skripsi.

7. Keluarga penulis terutama Ibu Hamidah dan Abah Arwani Sukanto. Yang telah memberikan dorongan motivasi luar biasa sehingga penulis mampu sampai pada titik ini.
8. Teruntuk kakak penulis, Ayu Nita Agustini. Yang telah mendukung dengan penuh kasih sayang sebagi seorang kakak kepada sang penulis.
9. Untuk Maulidya Rahmah, terima kasih telah mendukung penulis hingga mampu menyelesaikan perjalanan yang amat panjang ini sehingga mampu mencapai apa yang di inginkan ketika pertama memulai perjalanan di Universitas ini.

Penulis menyadari bahwa skripsi ini masih memiliki kekurangan, sehingga masukan, kritik, dan saran yang membangun dari pembaca sangat diharapkan untuk perbaikan di masa mendatang. Semoga karya ini dapat bermanfaat bagi para pembaca sekalian serta memberikan kontribusi positif dalam pengembangan ilmu pengetahuan.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Pratama, Prasetyo Putra. 2025. **Identifikasi Tempat Pengambilan Foto Jalan Di Kota Malang Menggunakan Metode Neural Network.** Skripsi. Program Studi Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Dr. Cahyo Crysdian, M.Cs (II) Dr. Irwan Budi Santoso, M.Kom.

Kata kunci: Neural Network, klasifikasi gambar, Kota Malang, identifikasi lokasi, K-Fold

Penelitian ini mengkaji identifikasi lokasi pengambilan foto jalan di Kota Malang menggunakan metode Neural Network. Tantangan dalam identifikasi lokasi dari citra digital seperti pencahayaan, sudut pengambilan, dan kemiripan visual antar jalan, menjadikan pendekatan manual kurang efektif. Oleh karena itu, pendekatan otomatis melalui Neural Network dipilih karena kemampuannya mengenali pola-pola kompleks dalam data visual. Data diambil dari lima ruas jalan di Kota Malang dan diproses dengan tahapan resizing, ekstraksi nilai RGB, dan pelatihan model menggunakan teknik K-Fold Cross Validation. Hasil penelitian menunjukkan bahwa model mampu mengklasifikasikan beberapa jalan dengan akurasi yang cukup baik, meskipun terdapat kelas jalan tertentu yang sering salah diklasifikasi. Model dengan K=3 menunjukkan performa lebih optimal dibandingkan K=5. Penelitian ini diharapkan memberikan kontribusi dalam pengembangan teknologi identifikasi lokasi untuk dokumentasi wisata, pengelolaan kota, dan promosi destinasi secara otomatis.

ABSTRACT

Pratama, Prasetyo Putra. 2025. Identification of Street Photo Locations in Malang City Using Neural Network Method. Undergraduate Thesis. Informatics Engineering Study Program, Faculty of Science and Technology, Maulana Malik Ibrahim State Islamic University of Malang. Supervisor: (I) Dr. Cahyo Crysdian, M.Cs (II) Dr. Irwan Budi Santoso, M.Kom.

Key words: Neural Network, image classification, Malang City, location identification, K-Fold.

This study investigates the identification of street photo locations in Malang City using the Neural Network method. Challenges in identifying locations from digital images such as lighting, shooting angles, and visual similarities between roads make manual approaches less effective. Therefore, an automatic approach through Neural Networks was chosen due to its ability to recognize complex patterns in visual data. The data were collected from five streets in Malang City and processed through resizing, RGB value extraction, and model training using K-Fold Cross Validation. The results showed that the model could classify several streets with reasonably good accuracy, although some specific classes were often misclassified. The model with K=3 demonstrated better performance compared to K=5. This research is expected to contribute to the development of location identification technologies for tourism documentation, urban management, and automated destination promotion.

البحث مستخلص

.براتاما، براسيتيو بوترا. 2025. تحديد مواقع التقاط صور الطرق في مدينة مالانج باستخدام طريقة الشبكات العصبية. أطروحة تخرج برنامج دراسة تقنية المعلومات، كلية العلوم والتكنولوجيا، الجامعة الإسلامية الحكومية مولانا مالك إبراهيم مالانج. المشرفون الدكتور كاهيو كريسديان، ماجستير في علوم الحاسوب، الدكتور إروان بودي سانتوسو، ماجستير في علوم الحاسوب.

الكلمات المفتاحية: لشبكات العصبية، تصنيف الصور، مدينة مالانج، تحديد المواقع، التحقق المتقاطع

تتناول هذه الدراسة تحديد موقع التقاط صور الطرق في مدينة مالانج باستخدام طريقة الشبكات العصبية. التحديات في تحديد المواقع من الصور الرقمية مثل الإضاءة، زاوية الالتقاط، والتشابه البصري بين الطرق تجعل النهج اليدوي غير فعال. لذلك، تم اختيار النهج الآلي من خلال الشبكات العصبية بسبب قدرتها على التعرف على الأنماط المعقدة في البيانات البصرية. تم جمع البيانات وتدريب النموذج باستخدام تقنية ،RGB من خمسة طرق في مدينة مالانج وتمت معالجتها بمراحل تغيير الحجم، واستخلاص قيم أظهرت نتائج الدراسة أن النموذج قادر على تصنيف بعض الطرق بدقة جيدة، على الرغم من وجود .K-Fold التحقق المتقاطع ومن المتوقع أن تسهم .K=5 أداءً أكثر كفاءة مقارنة بـ K=3 بعض الطرق التي تم تصنيفها بشكل خاطئ. أظهر النموذج باستخدام .هذه الدراسة في تطوير تقنيات تحديد المواقع لأغراض التوثيق السياحي، وإدارة المدن، والترويج التلقائي للمواقع السياحية

BAB I

PENDAHULUAN

1.1 Latar Belakang

Kota Malang di Jawa Timur dikenal sebagai salah satu tujuan wisata utama di Indonesia, menarik wisatawan lokal dan mancanegara dengan keindahan alam, kekayaan budaya, dan kulinernya yang khas. Beragam destinasi seperti pegunungan, pantai, serta objek wisata sejarah dan budaya menjadi daya tarik utama kota ini, yang tentunya sering diabadikan dalam berbagai foto oleh pengunjung (Ghassani, 2022).

Mengidentifikasi lokasi suatu jalan dari sebuah foto merupakan tantangan yang kompleks. Faktor-faktor seperti variasi pencahayaan, sudut pengambilan gambar, dan kemiripan visual antara satu jalan dengan jalan lainnya seringkali menyulitkan proses identifikasi secara manual (Putri, 2023). Mengandalkan pengamatan manusia dalam proses ini tidak hanya memakan waktu, tetapi juga berisiko tinggi terhadap kesalahan, terutama ketika harus menelusuri ribuan foto yang diambil di berbagai tempat. Sebagai contoh, dalam pengelolaan data geospasial di Badan Informasi Geospasial (BIG), ketepatan lokasi merupakan aspek yang krusial untuk mendukung perencanaan pembangunan infrastruktur di Kota Malang (Fernando, 2020).

Berbagai metode telah digunakan untuk mengatasi masalah ini. Metode manual yang mengandalkan pengamatan manusia adalah yang paling tradisional, namun terbukti tidak efisien dan rentan terhadap kesalahan. Metode berbasis

pencocokan fitur seperti SIFT (*Scale-Invariant Feature Transform*) dan SURF (*Speeded Up Robust Features*) menawarkan hasil yang lebih baik, tetapi tetap tidak optimal dalam kondisi gambar yang tidak ideal. Selain itu, ada juga metode pencocokan gambar menggunakan GIS (*Geographic Information System*) yang memerlukan basis data spasial yang besar dan komputasi intensif, menjadikannya tidak praktis untuk diterapkan dalam skala besar.

Dengan penerapan sistem berbasis Neural Network, tantangan ini diharapkan dapat diatasi dengan lebih efisien. Neural Network memiliki kemampuan untuk mengenali pola dan karakteristik unik dalam gambar yang mungkin tidak terlihat jelas oleh mata manusia. Teknologi ini memungkinkan sistem untuk mengidentifikasi lokasi asal sebuah foto secara otomatis dan akurat. Penerapan teknologi seperti ini dapat mendukung berbagai program pemerintah, seperti pengelolaan informasi pariwisata oleh Kementerian Pariwisata dan Ekonomi Kreatif, serta pengarsipan dan promosi yang lebih efektif dan tepat sasaran dalam upaya meningkatkan daya tarik destinasi wisata pada Kota Malang (Pratama, 2022).

Teknologi *Artificial Intelegence* (AI), khususnya neural network, menawarkan solusi untuk mengatasi masalah ini dengan otomatisasi dan efisiensi (Azmi, 2023). Neural network mampu mengenali pola dan fitur spesifik dari gambar, memungkinkan klasifikasi lokasi foto secara lebih akurat dibandingkan metode manual. Teknologi ini telah terbukti efektif dalam pengenalan pola yang rumit, bahkan dengan variasi pencahayaan dan sudut pengambilan gambar (R. Fathur, 2022).

Penerapan neural network untuk klasifikasi lokasi foto dapat dimanfaatkan dalam berbagai hal, termasuk dokumentasi tempat wisata, pengelolaan aset kota, dan promosi pariwisata yang lebih efektif (Nugroho, 2022). Misalnya, dengan mengenali lokasi spesifik dari foto yang diunggah oleh wisatawan, strategi promosi dapat disesuaikan berdasarkan tren lokasi yang sering difoto (Devi, 2024).

Penelitian ini fokus pada pengoptimalan parameter neural network untuk meningkatkan akurasi klasifikasi lokasi pada gambar. Pengaturan parameter yang tepat sangat penting agar model Neural Network bekerja optimal, mengatasi variasi kondisi gambar, dan memberikan hasil yang bermanfaat bagi pengembangan pariwisata Kota Malang.

Dalam perspektif Islam, penelitian ini sejalan dengan prinsip-prinsip keilmuan dan kemajuan teknologi yang dianjurkan dalam ajaran Islam. Islam mendorong umatnya untuk berusaha dengan bersungguh-sungguh dalam mencari keridhaan Allah dan manfaat bagi masyarakat. Sebagaimana Allah SWT berfirman dalam Al-Qur'an Surat Al-Ankabut [29:69]:

وَالَّذِيْنَ جَاهَدُوْا فِيْنَا لَنَهْدِيَنَّهُمْ سُبُلَنَاۗ وَاِنَّ اللّٰهَ لَمَعَ الْمُحْسِنِيْنَ

"Dan orang-orang yang berusaha dengan bersungguh-sungguh untuk (mencari keridhaan) Kami, benar-benar akan Kami tunjukkan kepada mereka jalan-jalan Kami. Dan sesungguhnya Allah benar-benar bersama orang-orang yang berbuat baik"(QS.Al-Ankabut[29:69]).

Siapa yang dimaksud pejuang Allah. Ibnu Katsir menjelaskan bahwa “orang-orang yang berjihad untuk (mencari keridhaan) Kami” terutama merujuk kepada Nabi Muhammad SAW beserta para sahabat dan pengikutnya hingga akhir zaman. Mereka adalah contoh nyata mujahid yang bersungguh-sungguh di jalan Allah (Tafsir Ibnu Katsir).

1.2 Pernyataan Masalah

Bagaimana melakukan optimalisasi untuk identifikasi asal sebuah gambar jalan di Kota Malang menggunakan metode Neural Network (NN)?

1.3 Batasan Masalah

Berikut adalah batasan masalah yang di gunakan dalam penelitian ini:

a. Penelitian ini membatasi resolusi gambar yang digunakan untuk klasifikasi menjadi 60 x 60 piksel, di mana foto-foto akan di-*resize* ke ukuran tersebut sebelum proses klasifikasi, meskipun diakui bahwa batasan ini dapat mengurangi detail visual pada gambar. Foto-foto akan di-*resize* menjadi ukuran tersebut sebelum proses klasifikasi.
b. Semua gambar ada di dalam area kota malang meliputi Jalan Golf Kayu Tangan, Jalan Kayutangan, Jalan Veteran, Jalan Suhat dan Jalan Ijen.
c. Gambar yang di gunakan dalam proses training di ambil dari jam 10.00 WIB – 14.00 WIB dengan cuaca sedang cerah.

1.4 Tujuan Penelitian

Tujuan utama dari penelitian ini adalah untuk mengoptimalkan model Neural Network (NN) yang dirancang khusus untuk mengidentifikasi lokasi pengambilan foto yang berada di jalan-jalan Kota Malang. Model ini diharapkan dapat menghasilkan tingkat akurasi yang tinggi dalam menentukan lokasi tersebut dengan mempertimbangkan berbagai variabel yang mungkin mempengaruhi, seperti kondisi jalan, fitur geografis, dan elemen-elemen lain yang dapat dikenali dalam foto.

1.5 Manfaat Penelitian

Berikut adalah manfaat dari dilakukannya penelitian ini, terdapat manfaat teoritis dan manfaat praktis, yaitu:

a. Sistem ini dapat digunakan oleh Instansi Pemerintahan Negara Republik Indonesia, untuk membantu mengidentifikasi gambar.
b. Turis yang sedang berkunjung ke Kota Malang juga dapat menggunakan sistem ini untuk mengetahui nama jalan dari gambar.

BAB II

STUDI PUSTAKA

Untuk membantu penulis memahami lebih dalam tentang tema-tema yang berkaitan dengan perkembangan penelitian ini, dilakukan tinjauan literatur. Tinjauan tersebut menjelaskan studi-studi sebelumnya yang digunakan sebagai referensi dan analogi yang akan dibuat.

Penelitian Amarullah (2023), mengkaji perubahan tutupan lahan di Kota Malang dari tahun 2015 hingga 2023, dengan fokus pada peningkatan lahan pemukiman dan penurunan vegetasi serta badan air. Menggunakan model Cellular Automata (CA) dan Artificial Neural Network (ANN), penelitian ini memprediksi perubahan tutupan lahan di masa depan dengan akurasi 90% dan indeks kappa 83%, sejalan dengan RTRW hingga tahun 2035.

Penelitian Muhammad Gusanwa Akbar (2023), menggunakan kecerdasan buatan, khususnya Neural Network, untuk mengoptimalkan Barang Milik Daerah dengan mengklasifikasi fungsi aset. Penelitian ini mengintegrasikan data aset, sistem informasi, dan kecerdasan buatan untuk mendukung pengambilan keputusan. Pengujian menunjukkan peningkatan akurasi model dari 81% menjadi 95%, yang membuktikan efektivitas integrasi ini dalam pengelolaan aset.

Penelitian Azzah Roudhoh (2023), menggunakan Convolutional Neural Network (CNN) untuk mengklasifikasi daun sirih dan daun binahong berdasarkan bentuknya. Untuk mempercepat proses training yang memakan waktu lama dengan CPU, penelitian ini menggunakan GPU dengan CUDA. Hasilnya, GPU

mempercepat waktu training hingga dua kali lebih cepat dibandingkan CPU menggunakan 900 data *image* yang dibagi dalam rasio 8:1:1 untuk *training*, *testing*, dan *validation*.

Penelitian yang dilakukan oleh E. Setiadi dan A. Wibowo (2023), meneliti trotoar sebagai bagian jalan yang disediakan untuk pejalan kaki, namun sering kali mengalami kerusakan yang mengganggu dan membahayakan pengguna. Perbaikan trotoar diawali dengan mendeteksi kerusakan, salah satunya melalui metode deep learning menggunakan Convolutional Neural Network (CNN). Penelitian ini bertujuan membedakan trotoar retak dan tidak retak dengan dataset latih sebanyak 4000 gambar dari Kaggle, serta dataset uji gabungan dari Kaggle dan data internal. Hasil pengujian menunjukkan model mampu mengklasifikasikan kondisi trotoar dengan akurasi tinggi, rata-rata di atas 96%.

Penelitian yang dilakukan oleh Radikto, Dadang Iskandar Mulyana, M. Ainur Rofik, dan M. Ohan Zoharuddin Zakaria (2022), meneliti pengenalan citra kendaraan menggunakan algoritma Convolutional Neural Network (CNN). Penelitian ini mengembangkan model pengenalan gambar kendaraan seperti sepeda, sepeda motor, mobil, bus, dan truk, menggunakan dua model uji: *Sequential* dan VGG16. Dengan 1406 data latih dan 274 data uji, model Sequential mencapai akurasi 98,18% dan loss 0,103, sedangkan model VGG16 menghasilkan akurasi 99,64% dengan loss 0,014.

Penelitian yang dilakukan oleh Yusup Yulianto dan Ari Wibowo (2022), meneliti kondisi jalan yang mempengaruhi kenyamanan pengguna. Dengan menggunakan algoritma Convolutional Neural Network (CNN), penelitian ini

bertujuan mendeteksi kerusakan jalan melalui input gambar permukaan dari Kaggle sebanyak 2074 citra. Proses preprocessing mencakup konversi citra RGB ke *grayscale*, *resize*, dan *edge detection*, yang diikuti oleh *training* dataset. Hasil pengujian menunjukkan sistem berhasil mencapai akurasi 92,9%.

Penelitian oleh Ni Nyoman Citariani Sumartha, I Gede Pasek Suta Wijaya, Fitri Bimantoro, dan Gibran Satya Nugraha (2024), berfokus pada lubang di jalan (pothole) yang membahayakan keselamatan pengemudi. Gambar lubang memiliki variasi dalam kontras warna, ukuran, dan pencahayaan, memerlukan metode identifikasi yang efektif. Penelitian ini menggunakan Convolutional Neural Networks (CNN) untuk mengenali foto lubang jalan. Hasilnya menunjukkan akurasi tertinggi mencapai 0,99 dengan ukuran gambar 128x128, tiga lapisan tersembunyi, dan 128 neuron.

Penelitian yang dilakukan oleh Muhammad Mahrus Ali, Muhammad Faishol, dan Khoirul Anwar (2022), mengkaji kerusakan aspal sebagai faktor utama penentu kelayakan jalan, yang memerlukan pemeliharaan berkala. Untuk menghindari gangguan lalu lintas dan risiko keselamatan, penelitian ini mengembangkan sistem deteksi jalan berlubang menggunakan video dengan metode Gray Level Co-occurrence Matrix (GLCM) dan Neural Network. Proses deteksi terdiri dari ekstraksi ciri citra dan pelatihan manual. Hasil pengujian menunjukkan Recall 0,80, Precision 0,06, Accuracy 0,79, dan Error Rate 0,20.

Penelitian oleh Maharany Shandra Ayu Hapsary, Sawitri Subiyanto, dan Hana Sugiastu Firdaus (2021) menganalisis perubahan penggunaan lahan di Kota Balikpapan dari 2009 hingga 2019, serta memprediksi untuk tahun 2029

menggunakan model Artificial Neural Network (ANN) dan Regresi Logistik. Data berasal dari citra Landsat 7, SPOT 5, dan SPOT 7. Hasilnya menunjukkan penurunan kebun campuran sebesar 3.499,69 Ha (6,85%) dan peningkatan mangrove sebesar 2.515 Ha (4,92%). Model ANN memiliki akurasi lebih tinggi dengan indeks kappa 0,620, sementara kesesuaian peta prediksi tahun 2029 terhadap RTRW adalah 44,25% untuk ANN dan 43,49% untuk regresi logistik.

BAB III

DESAIN DAN IMPLEMENTASI

3.1 Pengumpulan Data

Dalam penelitian ini, data yang digunakan berupa gambar digital yang diperoleh melalui pengambilan gambar menggunakan perangkat ponsel. Lokasi pengambilan gambar dilakukan di beberapa titik di Kota Malang, yaitu Jalan Golf Kayu Tangan, Jalan Kayu Tangan, Jalan Suhat, Jalan Ijen dan Jalan Veteran. Pemilihan lokasi ini didasarkan pada variasi kondisi lingkungan yang diharapkan dapat mempengaruhi spektrum warna yang terekam pada citra. Penggunaan perangkat ponsel sebagai alat pengambilan gambar dipilih karena pertimbangan kemudahan akses dan portabilitas, sehingga memungkinkan pengumpulan data dilakukan secara efisien. Hasil dari gambar yang di ambil mempunyai ukuran 3056 x 3056 pixel, ukuran ini adalah ukuran tertinggi yang bisa di ambil menggunakan perangkat yang ada.

Adapun beberapa rincian teknis terkait pengambilan dan pengolahan gambar sebagai berikut:

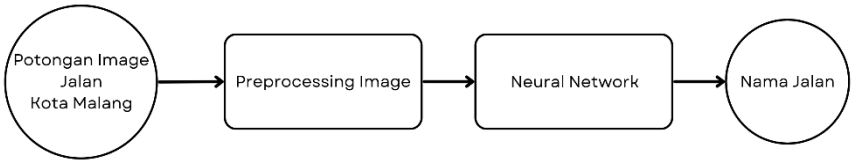
a. Alat pengambilan gambar: Gambar diambil menggunakan ponsel Realme GT Master Edition yang dilengkapi dengan kamera beresolusi 64 MP dan prosesor Qualcomm Snapdragon 778G Octa-Core.
b. Gambar berwarna: Gambar yang diambil merupakan gambar berwarna dengan spektrum warna RGB.

c. Lokasi pengambilan: Foto diambil di lima lokasi berbeda, yaitu Jalan Golf Kayu Tangan, Jalan Kayu Tangan, Jalan Suhat, Jalan Ijen dan Jalan Veteran semuanya di dalam area Kota Malang.

d. Waktu pengambilan: Pengambilan gambar dilakukan pada pagi hari mulai dari jam 10.00 - 14.00 WIB, tepatnya pada tanggal 1 Desember 2024.

3.2 System Design

Sistem ini dirancang agar mampu menangani data citra yang diperoleh dan mengkonversinya menjadi bentuk yang dapat digunakan oleh algoritma pembelajaran mesin. Desain sistem ini juga mencakup struktur alur kerja yang diimplementasikan dalam tahapan-tahapan yang jelas.



Gambar 3.1 Sistem Design

Pada gambar 3.1 tahapan dimulai dari Potongan Image Kota Malang yang dimana telah di kumpulkan sebelumnya. kemudian image akan di lakukan tahap *Preprocessing* yaitu tahap *resizeing* dan *extraction.* Lalu akan di lanjutkan pada tahap Neural Network untuk di lakukan training dari image yang telah di *preprocessing* dan kemudian akan menghasilkan output berupa nama jalan, berikut adalah penjelasan lebih lengkapnya.

3.2.1 Potongan *Image* Jalan Kota Malang

Image Jalan Kota Malang selanjutnya akan digunakan sebagai masukan utama dalam sistem yang telah dikembangkan untuk tujuan penelitian ini. Gambar-

gambar jalan yang diambil di berbagai lokasi di Kota Malang akan diolah untuk diekstraksi informasi pentingnya.

Gambar 3.2 Potongan Image Jalan Kota Malang

Dengan memanfaatkan perangkat keras dan perangkat lunak yang telah dijelaskan secara rinci pada subbab 3.1, diharapkan bahwa gambar-gambar tersebut dapat diubah menjadi data yang berkualitas tinggi dan siap untuk diproses lebih lanjut. Proses pengolahan ini bertujuan agar gambar yang diperoleh dapat memberikan representasi yang akurat dan relevan terhadap kondisi jalan di Kota Malang, sehingga dapat mendukung kinerja sistem yang telah dibuat dengan optimal.

3.2.2 *Preprocessing Image*

Image yang sudah dimasukkan ke dalam sistem kemudian akan di proses, proses yang terjadi melibatkan beberapa tahap yang pertama ada *resizing*, ekstrak RGB dan *convert image*.

a. Pada tahap pertama akan terjadi proses *resize* yang awalnya mempunyai ukuran 3056 x 3056 pixel menjadi 60 x 60 pixel menggunakan code sistem yang telah di buat. Ini bertujuan untuk memudahkan proses training dengan data yang lebih ringan, berikut adalah contoh image sebelum resizing dan sesudah resizing.

Gambar 3.3 Potongan Jalan Golf Kayu Tangan Asli

Gambar 3.4 Potongan Jalan Golf Kayu Tangan 60 x 60

Pada gambar 3.2 bisa di lihat adalah gambar asli sebelum dilakukan proses resizing yang dimana ukuran awal image adalah 3056x3056 pixel. kemudian pada gambar 3.3 adalah hasil dari resizing menjadi ukuran 60x60 pixel.

b. Lalu setelah itu image yang telah di *resize* akan di ekstrak nilai RGB nya dengan cara reading image menggunakan kode program yang telah di buat untuk membaca nilai RGB yang ada pada *Image*. Proses ini akan memisahkan nilai RGB pada pixel pixel yang ada menjadi 3 nilai, yaitu nilai *Red*, *Green* dan *Blue*.

c. Kemudian pada tahap terakhir nilai tersebut di masukan kedalam file csv untuk setiap kategorinya. Terdapat 3 file kategori, yaitu merah (*red*), hijau (*green*) dan biru (*blue*). Total data pada setiap filenya adalah 1000 baris data dengan jumlah kolom 3600 kolom. Jika di jumlahkan total nilai yang akan

di masukan kedalam Neural Network adalah 3.600.000 nilai untuk setiap filenya, jika dijumlahkan total dari 3 file adalah 10.800.000 nilai.

Tahap convert dari nilai RGB ke dalam file CSV adalah tahap terakhir dari preprocessing, selanjutnya data file akan di kirim ke sistem Neural Network.

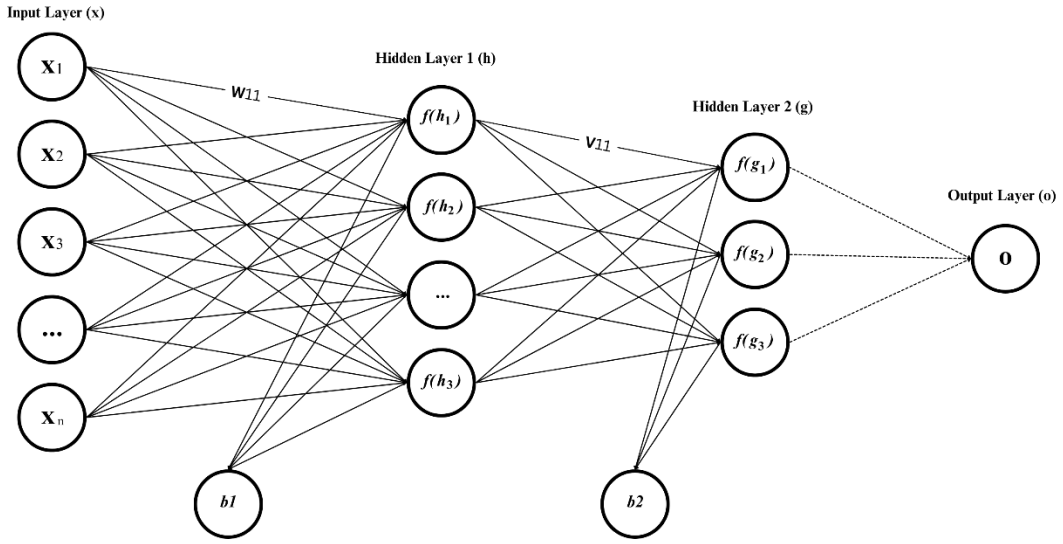
3.2.3 Neural Network

Pada bagian ini, pembahasan mengenai Neural Network akan dijelaskan secara rinci dan terbagi menjadi tiga subbagian utama, yaitu Desain, Training, dan Eksperimen. Setiap subbagian ini akan menggali lebih dalam mengenai proses pengembangan model Neural Network yang digunakan dalam penelitian ini. Pada subbagian Desain, akan dibahas mengenai struktur dan arsitektur dari model Neural Network yang dipilih, termasuk pemilihan jenis lapisan dan parameter yang digunakan. Selanjutnya, pada subbagian Training, akan dijelaskan mengenai proses pelatihan model, teknik yang digunakan untuk mengoptimalkan hasil, serta dataset yang digunakan untuk melatih model tersebut. Terakhir, pada subbagian eksperimen, akan dibahas mengenai serangkaian eksperimen yang dilakukan untuk menguji performa dan akurasi model Neural Network yang telah dibangun, serta analisis terhadap hasil yang diperoleh dari eksperimen-eksperimen tersebut.

3.2.3.1 Desain Neural Network

Pada tahap ini, sistem Neural Network akan menerima value data RGB dari 3 file yang sudah ada, kemudian data tersebut akan disatukan yang menjadi sebuah array. Array tersebut mempunyai urutan R, G, dan B yang berurut dari masing masing file, proses ini disebut dengan *flatten*.

Setelah itu, model akan menjalani proses pelatihan menggunakan metode *k-fold cross-validation*. *K-fold cross-validation* adalah teknik evaluasi model yang membagi dataset menjadi *k* bagian atau *folds*. Model akan dilatih sebanyak *k* kali dengan setiap iterasi menggunakan *k-1* bagian sebagai data latih dan satu bagian sebagai data uji, sehingga seluruh data dapat digunakan untuk pelatihan dan pengujian secara bergantian. Proses ini membantu mengurangi risiko *overfitting* dan memberikan evaluasi performa model yang lebih akurat, kemudian melakukan proses neural network sesuai Gambar 3.5 berikut.



Gambar 3.5 *Neural Network*

Proses dimulai dari data input yang berupa representasi nilai dari sebuah *pixel* yang dimana digunakan sebagai informasi awal dari Neural Network. *Hidden layer* berada di antara *input layer* dan *output layer* yang berfungsi untuk memetakan sebuah informasi yang ada pada internal. Pada Neural Network akan dilakukan perhitungan berupa hasil kali dari input dengan bobot, kemudian fungsi aktivasi di terapkan untuk mendapatkan *output* dari *layer* sebelumnya dan kemudian akan dikirimkan sebagai input pada *layer* berikutnya.

a. *Input Layer* : Panjangnya sesuai dari ukuran gambar

b. *Hidden Layer 1* : Jumlah *node* mengikuti jumlah skenario pengujian

c. *Hidden Layer 2* : Mengikuti jumlah *class* yang ada yaitu 3 buah *class*

d. *Output Layer* : Menggunakan 1 *node* (salah satu dari 3 *class*)

Kemudian setiap *pixel* dari masing masing kategori akan dilakukan pengalian antara input (*pixel*) dengan bobot, berikut ini adalah rumus untuk perkalian antara input dan bobot pada persamaan 3.1:

$$w_i = \sum_{j=1}^{n} (x_i \times w_{ji}) + b_1 \quad (3.1)$$

Keterangan:
x_i = *input*
w_{ji} = *bobot*
b_1 = *bias*
w = *output yang dihasilkan sebelum masuk kedalam fungsi aktivasi ReLU*

Lalu pada *hidden layer 1* akan menampung hasil dari sebuah perkalian nilai dari input dengan bobot yang tersambung dari neuron pada *input layer* dengan *hidden layer 1*. Lalu hasil dari perkalian tadi akan diaktivasi menggunakan fungsi ReLu, persamaan 3.2 merupakan fungsi aktivasi ReLu.

$$h_i = max(0, w_i) \quad (3.2)$$

Keteragan:
w_i = *input dari fungsi relu*

ReLU (*Rectified Linear Unit*) adalah fungsi aktivasi yang digunakan untuk memperkenalkan non-linearitas ke dalam jaringan saraf. Tanpa fungsi aktivasi, jaringan saraf hanya akan menjadi kombinasi linier dari input, yang artinya tidak akan bisa memecahkan masalah yang kompleks atau non-linear.

Kemudian setelah itu hasil dari aktivasi ReLU akan dijadikan sebagai input pada perhtungan bobot *v*.

$$v_{\mathrm{i}} = \sum_{\mathrm{j}=1}^{\mathrm{n}} \left(h_{\mathrm{i}} \times w_{\mathrm{ji}}\right) + \mathrm{b}_2 \tag{3.3}$$

Keterangan:
h_i = *input*
b_2 = *bias*

Lalu dilakukan perhitungan *softmax* untuk mengubah nilai menjadi probabilitas berkisar 0 -1, berikut adalah rumus untuk perhitungan *softmax* sebagai *hidden layer 2*.

$$o_i = \frac{\exp(h_i)}{S} \tag{3.4}$$

Keteragan:
$\hat{o}$ = *hasil output dari model neural network*

Dimana s merupakan total eksponensial menggunakan persamaan 3.5.

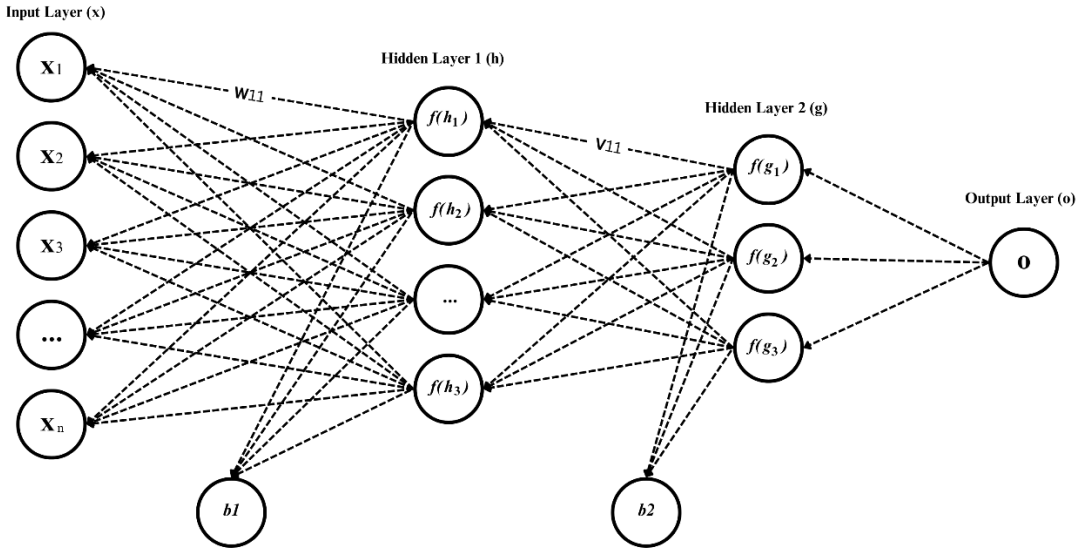
$$S = \sum_{j=1}^{n} \exp\,(h_i) \tag{3.5}$$

Setelah mendapatkan nilai probability dari setiap kelas yang ada, selanjutnya akan dilakukan perhitungan pengambilan *probability* terbesar menggunakan persamaan 3.6.

$$\hat{o} = \mathrm{argmax}(\, o_i) \tag{3.6}$$

3.2.3.2 *Training*

Tahapan yang pertama kali dilakukan pada model adalah training. Pada tahap ini, model neural network dilatih menggunakan data training. Tujuanya adalah mendapatkan bobot beserta bias yang optimal untuk model nantinya, berikut merupakan alur dari *backpropagation* pada Gambar 3.6.



Gambar 3.6 *Backpropagation Neural Network*

Kemudian akan dilakukan perhitungan *gradient loss* untuk menentukan seberapa besar perubahan bobot yang harus dilakukan, berikut adalah tahapan tahapan untuk menghitung *gradient loss*.

Cross-Entropy Loss digunakan untuk mengukur seberapa jauh prediksi dari target sebenarnya.

$$E = -\sum_{j=1}^{3} y \log(\hat{o}) \tag{3.5}$$

Jumlah kelas (B) adalah 5, target kelasyang benar adalah kelas 2 (y: [0,1,0,0,0]), dan probabilitas *output* dari softmax (*hidden layer* 2) adalah ([0.0000743, 0.99, 0.0067, 0.0000451, 0.00000000827]). Karena hanya y_2 = 1 yang berpengaruh dalam Cross-Entropy Loss, maka:

$$E = -(1\ x\ log(0.99))$$

$$E = -(-0.01005) = 0.01005$$

Kemudian akan dilakukan *update* bobot untuk menghasilkan *training* yang lebih baik.

$$w_{baru} = w_{lama} - learning\_rate\ x\ E \tag{3.6}$$

Setelah dilakukan proses update bobot, maka proses selanjutnya adalah update *bias seperti persamaan 3.7*:

$$b_{baru} = b_{lama} - learning\_rate\ x\ E \tag{3.7}$$

3.2.3.3 *Testing*

Dalam proses pengujian model, dilakukan beberapa perhitungan metrik evaluasi guna menilai performa model secara menyeluruh. Metrik yang digunakan meliputi *Precision, Recall,* dan *F1-Score*, yang masing-masing memiliki peran penting dalam mengevaluasi hasil prediksi. Precision mengukur sejauh mana prediksi positif yang dilakukan oleh model benar adanya, sementara Recall mengukur sejauh mana model mampu menangkap seluruh data positif yang sebenarnya. F1-Score, sebagai rata-rata harmonis dari Precision dan Recall, memberikan gambaran seimbang terhadap kinerja model, berikut adalah rumus yang di gunakan untuk menghitung *Precision, Recall,* dan *F1-Score*;

$$Presisi = \frac{TP}{TP + FP} \tag{3.8}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.9}$$

$$F1 - score = \frac{2x\ Presisi\ x\ Recall}{Presisi + Recall} \tag{3.10}$$

BAB IV

UJI COBA DAN PEMBAHASAN

4.1 Skenario Pengujian

Dalam penelitian ini, data di ambil secara langsung di beberapa area pada kota malang dengan menggunakan alat sesuai pada pembahasan BAB 3.1, untuk rincian data akan ditunjukan pada Table 4.1.

Tabel 4.1 Data Jalan.

No.	Nama Jalan	Total Gambar
1.	Jalan Golf Kayu Tangan	200
2.	Jalan Kayu Tangan	200
3.	Jalan Veteran	200
4.	Jalan Ijen	200
5.	Jalan Suhat	200

Masing-masing dari jalan mempunyai 200 gambar sebagai data *training*, ini bertujuan agar semua data pada setiap jalan seimbang satu dengan yang lainya. Kemudian tahapan selanjutnya merupakan skenario pengujian yang bertujuan untuk mengevaluasi kinerja sistem, berikut adalah skenario pengujian yang dilakukan.

Tabel 4.2 Data Jalan.

No.	K Fold	Learning Rate	Minimal Loss	Tolerance
1	3	0.0001	0.1	0.001
2	5	0.0001	0.1	0.001

Pada table 4.2, terdapat 2 Skenario Pengujian, yang masing masing mempunyai Fold 3 dan 5. Kedua skenario tersebut menggunakan *learning rate* 0.0001, minimal *loss* 0.1 dan nilai *tolerance* 0.001.

4.2 Hasil Uji Coba

Kemudian dari Skenario Uji Coba pada BAB 4.1 terdapat skenario yang akan di lakukan untuk mendapatkan model terbaik, disana terdapat parameter *learning rate* dan *K fold* yang di *custom* agar mendapatkan beberapa variasi dengan tujuan mendapatkan hasil optimal.

Sementara untuk pembatasan agar pelatihan tidak mengalami stuck atu berulang maka di buat nilai batasan (min loss) degan nilai 0.1, jika nilai loss pelatihan sudah mencapai 0.1 maka pelatihan akan di selesaikan. Kemudian terdapat juga nilai minimal tolerance sebesar 0.001, ketika perubahan loss terakhir dengan saat ini kurang dari nilai tersebut maka pelatihan juga akan di selesakan. Pada tabel 4.3 merupakan hasil dari pelatihan secara garis besar.

Tabel 4.3 Hasil Pengujian.

No.	K Fold	Learning Rate	Minimal Loss	Tolerance	Avg Accurate
1	3	0.0001	0.1	0.001	0.453
2	5	0.0001	0.1	0.001	0.443

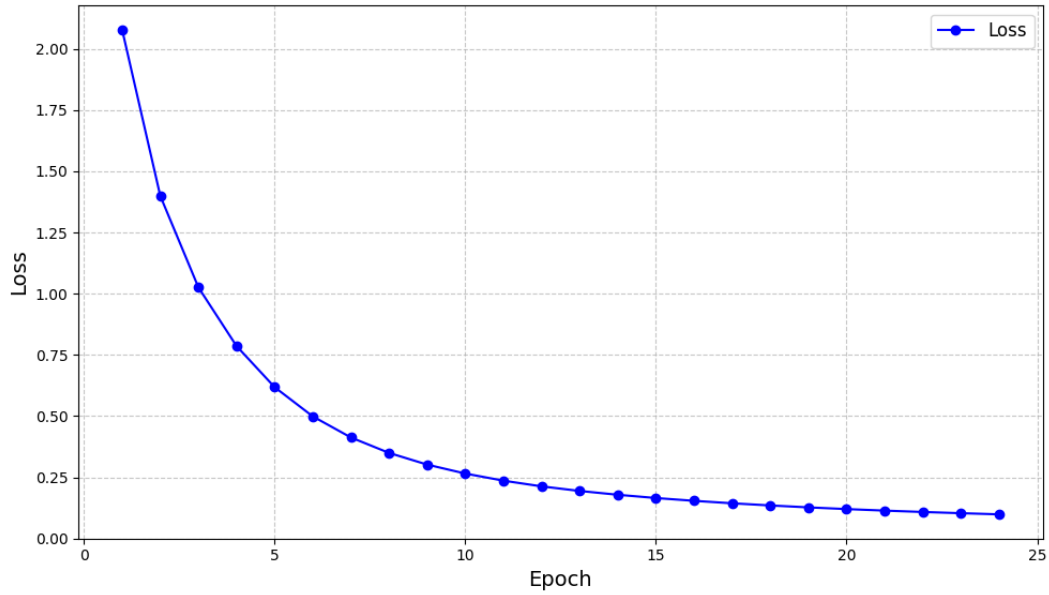
Pada tabel 4.3 menunjukkan hasil pengujian model dengan menggunakan teknik K-Fold Cross Validation dengan variasi jumlah fold (K-Fold = 3 dan 5). Parameter yang digunakan dalam pelatihan meliputi learning rate (0.0001), minimal loss (0.1), serta tolerance (0.001), sementara akurasi rata-rata (Avg Accurate) menjadi metrik evaluasi utama.

4.2.2 Training

Dalam proses training dilakukan menggunakan metode K fold 3 dengan learning rate 0.0001 dan total class yang adalah sejumlah 5 class yang meliputi

jalan_golf_kayu_tangan, jalan_kayu_tangan, jalan_veteran, jalan_ijen dan jalan_suhat.

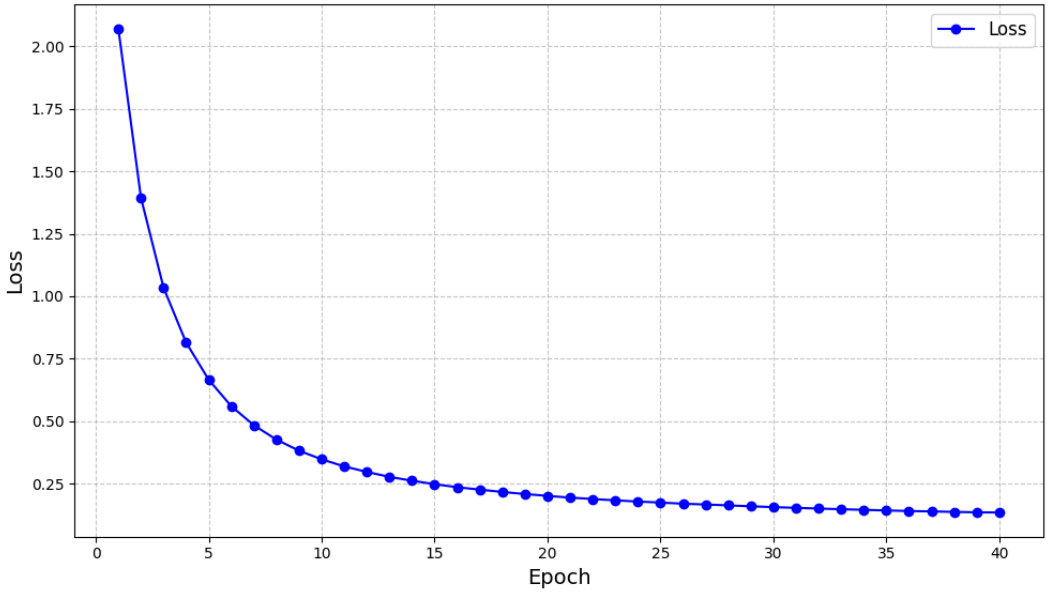
Pada fold 1 training berhenti di karenakan perubahan loss telah kurang dari 0.0010 dengan total epoch 24 dan loss 0.0986, berikut adalah grafik Loss Convergence untuk melihat bagaimana Loss bergerak dari awal epoch hingga akhir.



Gambar 4.1 Loss Convergence Fold 1

Pada gambar 4.1, terlihat bahwa nilai loss mengalami penurunan secara signifikan dari awal hingga akhir epoch, yang menunjukkan bahwa model berhasil mempelajari pola dari data dengan baik. Pada epoch pertama, nilai loss berada di atas 2.0, namun secara konsisten menurun seiring bertambahnya epoch dan mencapai nilai mendekati 0.1 pada epoch ke-24. Pola penurunan yang tajam di awal lalu melandai di akhir menandakan bahwa model mengalami proses learning yang stabil, di mana penyesuaian bobot semakin kecil seiring dengan mendekatinya nilai loss minimum. Hal ini mengindikasikan bahwa model memiliki potensi generalisasi yang baik terhadap data jika tren seperti ini juga konsisten di fold lainnya.

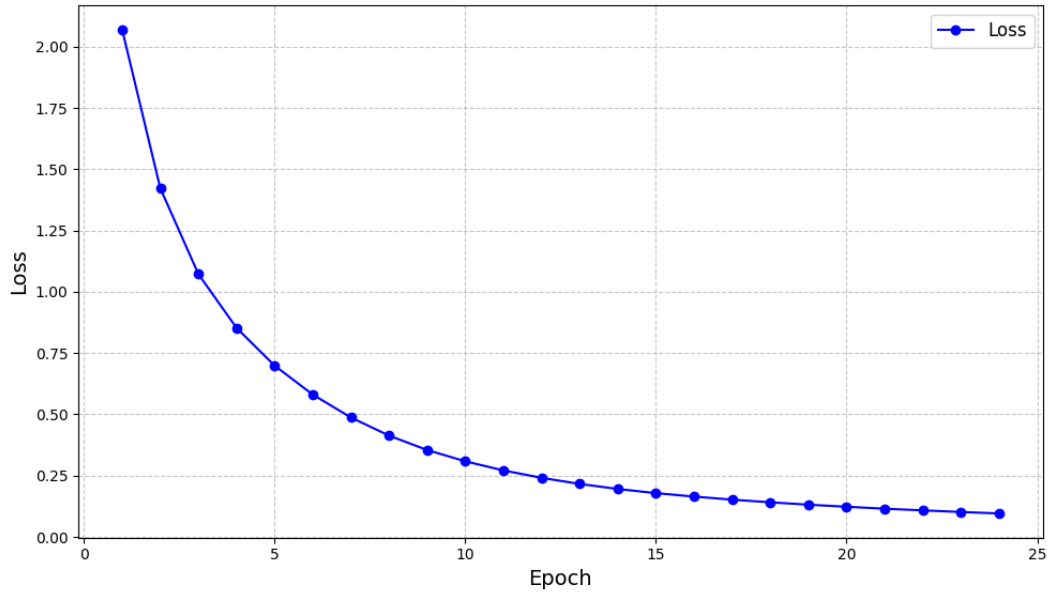
Pada fold 2 training berhenti di karenakan perubahan loss telah kurang dari 0.0010 dengan total epoch 40 dan loss 0.1342. Berikut adalah grafik Loss Convergence untuk melihat bagaimana Loss bergerak dari awal epoch hingga akhir.



Gambar 4.2 Loss Convergence Fold 2

Pada gambar 4.2, terlihat bahwa grafik di atas menggambarkan tren penurunan nilai loss selama proses training pada Fold 2. Nilai loss awal dimulai dari lebih dari 2.0 dan menunjukkan penurunan yang tajam hingga sekitar epoch ke-10. Setelah itu, penurunan menjadi lebih lambat namun tetap konsisten hingga mencapai nilai loss mendekati 0.15 pada epoch ke-40. Tren ini mengindikasikan bahwa proses pembelajaran pada model berlangsung secara efektif, di mana model mampu meminimalkan error secara bertahap dan stabil. Penurunan loss yang tidak fluktuatif juga menandakan bahwa proses training tidak mengalami overfitting atau ketidakseimbangan dalam pembelajaran, sehingga model memiliki kemampuan generalisasi yang cukup baik terhadap data validasi pada fold ini.

Pada fold 3 training berhenti di karenakan perubahan loss telah kurang dari 0.0010 dengan total epoch 24 dan loss 0.0957. Berikut adalah grafik Loss Convergence untuk melihat bagaimana Loss bergerak dari awal epoch hingga akhir.



Gambar 4.3 Loss Convergence Fold 3

Pada gambar 4.3, terlihat bahwa grafik di atas menunjukkan pola konvergensi loss selama proses training pada Fold 3. Nilai loss awal cukup tinggi, berada di atas 2.0, namun menunjukkan penurunan yang tajam pada epoch-epoch awal hingga sekitar epoch ke-10. Setelah itu, penurunan menjadi lebih gradual, mengikuti kurva eksponensial yang melandai, dan mendekati nilai stabil di bawah 0.15 menjelang epoch ke-24. Konsistensi penurunan tanpa lonjakan atau fluktuasi drastis mengindikasikan proses pelatihan yang stabil dan minim noise. Hal ini menunjukkan bahwa model mampu belajar secara progresif dan efisien dari data, serta memiliki potensi generalisasi yang baik tanpa indikasi overfitting pada fold ini, tabel 4.4 merupakan table detail dari skenario k fold 3.

Tabel 4.4 Detail Training K Fold 3.

No.	*Fold*	*Total Epoch*	*Loss*	Accurate
1	1	24	0.0986	0.4641
2	2	40	0.1342	0.4535
3	3	24	0.0957	0.4114

Pada tabel 4.4, menyajikan hasil pelatihan model menggunakan metode K-Fold Cross Validation dengan jumlah fold sebanyak 3. Teknik ini digunakan untuk mengukur performa model secara lebih akurat dengan membagi data latih ke dalam tiga bagian yang masing-masing bergantian berperan sebagai data validasi. Hasil evaluasi ditampilkan dalam bentuk jumlah epoch yang dibutuhkan, nilai loss, dan akurasi untuk setiap fold.

Pada Fold ke-1, proses pelatihan dilakukan selama 24 epoch dan menghasilkan nilai loss sebesar 0.0986 dengan tingkat akurasi 0.4641. Nilai ini menunjukkan bahwa model cukup mampu melakukan prediksi yang mendekati target dengan tingkat kesalahan (loss) yang relatif rendah.

Selanjutnya, pada Fold ke-2, jumlah epoch yang dibutuhkan meningkat signifikan menjadi 40 epoch. Meskipun pelatihan lebih lama, nilai loss justru naik menjadi 0.1342, dan akurasinya sedikit turun ke 0.4535. Hal ini bisa menandakan adanya overfitting atau kompleksitas data pada fold ini lebih tinggi, sehingga model kesulitan menemukan generalisasi yang baik.

Terakhir, Fold ke-3 menunjukkan hasil pelatihan selama 24 epoch dengan nilai loss paling rendah yaitu 0.0957, tetapi akurasi juga menjadi yang paling rendah yakni 0.4114. Ini mengindikasikan bahwa meskipun prediksi model mendekati nilai target (secara loss), klasifikasi yang dilakukan belum akurat dalam membedakan kelas dengan baik.

Secara keseluruhan, performa model dalam 3-fold ini menunjukkan bahwa nilai loss berada dalam rentang rendah, tetapi fluktuasi akurasi masih cukup tinggi. Hal ini menunjukkan bahwa model masih perlu disempurnakan untuk meningkatkan konsistensi akurasi prediksi pada semua fold. Rata-rata dari ketiga fold ini nantinya bisa digunakan sebagai tolok ukur kinerja model secara umum dalam proses validasi silang.

Tabel 4.5 Detail Training K Fold 5.

No.	***Fold***	***Total Epoch***	***Loss***	**Accurate**
1	1	46	0.113	0.475
2	2	54	0.0966	0.465
3	3	34	0.2239	0.41
4	4	21	0.293	0.46
5	5	26	0.0966	0.455

Pada tabel 4.5 menunjukkan hasil pelatihan model menggunakan teknik validasi silang K-Fold dengan K=5. Masing-masing baris pada tabel ini merepresentasikan hasil dari satu fold, termasuk informasi mengenai jumlah total epoch yang dibutuhkan, nilai loss, dan akurasi yang dicapai. Tujuan dari teknik ini adalah untuk menguji stabilitas dan generalisasi model terhadap data yang berbeda dengan membagi data pelatihan ke dalam lima subset.

Dari tabel 4.5 dapat dilihat bahwa Fold ke-2 memerlukan jumlah epoch terbanyak, yaitu sebanyak 54 epoch, dengan nilai loss terendah sebesar 0.0966, yang menandakan bahwa model pada fold ini cukup cepat mencapai konvergensi dan memberikan performa yang baik. Meskipun demikian, akurasinya (0.465) tidak jauh berbeda dari fold lainnya. Hal ini menunjukkan bahwa nilai loss yang rendah tidak selalu menjamin peningkatan akurasi yang signifikan.

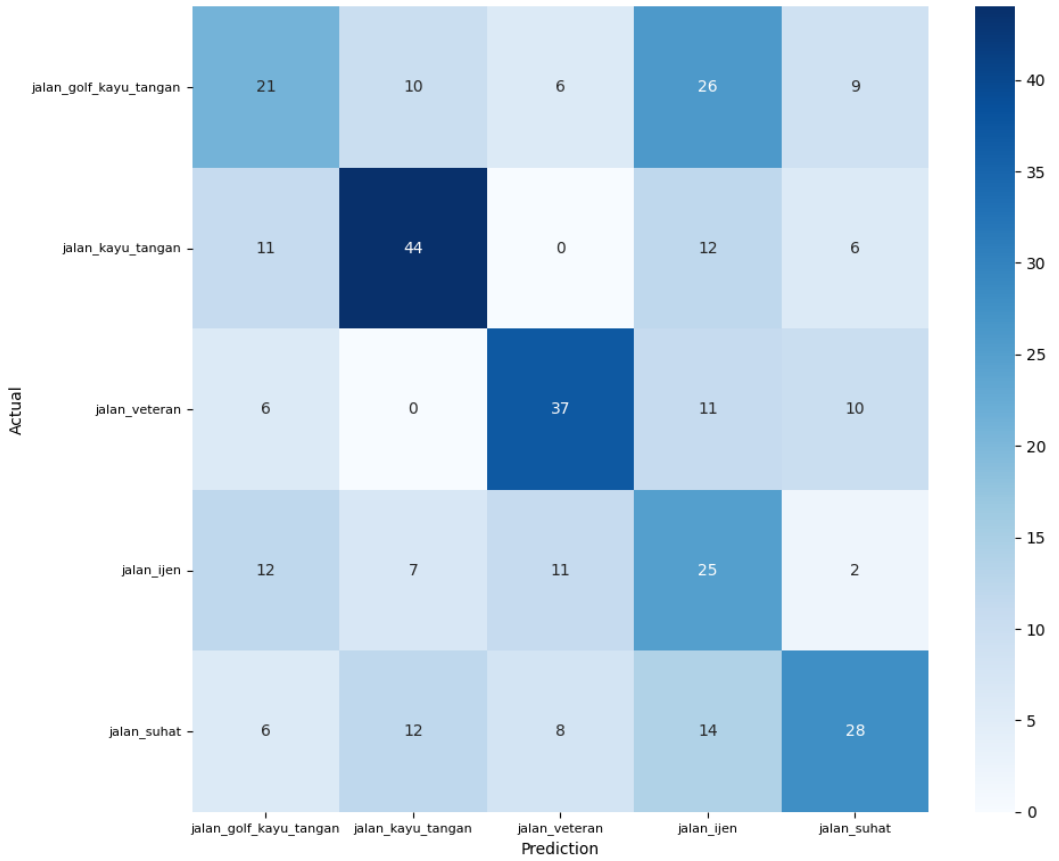
Sementara itu, Fold ke-4 hanya memerlukan 21 epoch namun memiliki nilai loss tertinggi, yaitu 0.293. Ini menunjukkan bahwa pelatihan pada fold ini mungkin terhenti lebih awal karena model tidak menunjukkan perbaikan signifikan, atau karena penerapan early stopping. Akurasi pada fold ini masih cukup kompetitif, yakni 0.46, yang mengindikasikan bahwa meskipun loss cukup tinggi, model masih mampu membuat prediksi yang cukup baik.

Secara keseluruhan, nilai akurasi berkisar antara 0.41 hingga 0.475, dengan rata-rata yang cukup konsisten. Hal ini menunjukkan bahwa model memiliki stabilitas yang baik dalam menghadapi variasi data pada tiap fold. Teknik K-Fold ini membantu memastikan bahwa model tidak terlalu bergantung pada subset data tertentu dan mampu melakukan generalisasi dengan cukup baik.

4.2.2 Testing

Dari hasil model yang sudah di training sebelumnya pada setiap fold maka akan di lakukan testing terhadap model tersebut. Berikut adalah hasil dari testing dari fold 1 sampai dengan 3 yang dimana modelnya di latih menggunakan learning rate 0.0001.

Pada fold 1 dengan hasil training total loss 0.0986 pada epoch terakhir di angka 24 mendapatkan akurasi testing sebesar 0.4641. Berikut merupakan Confusion Matrix dari fold 1.



Gambar 4.4 Confusion Matrix Model Testing Fold 1

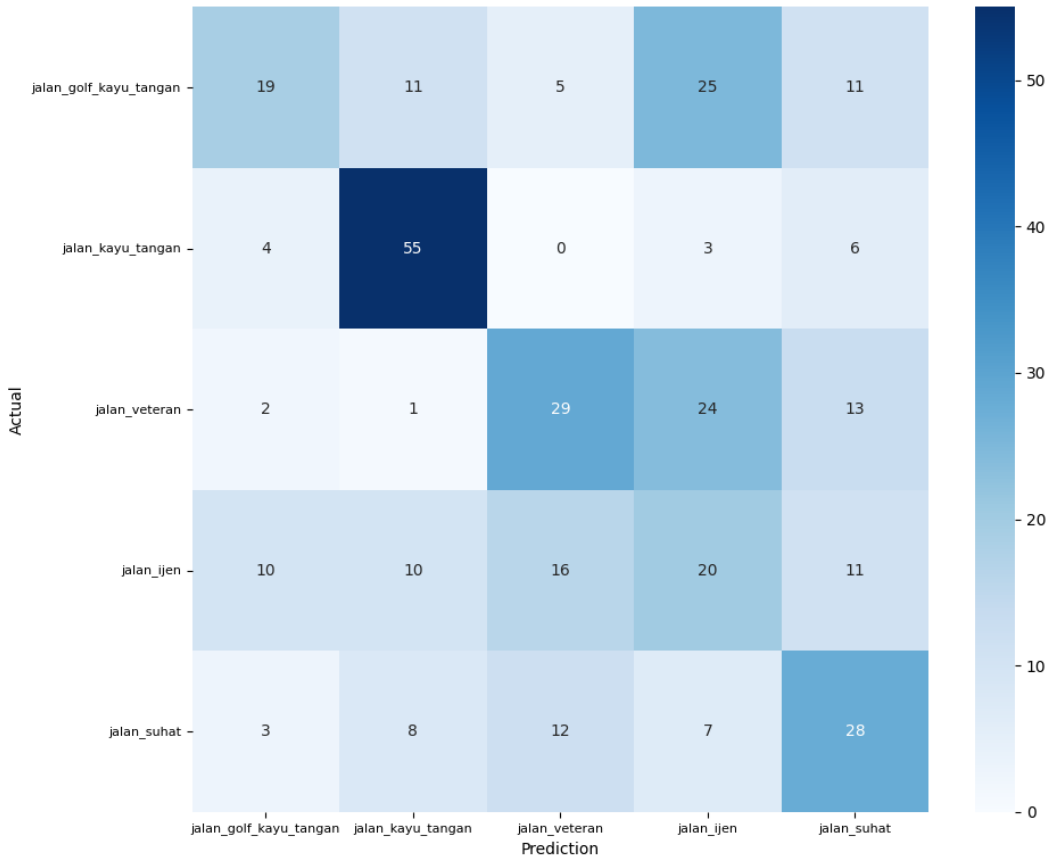
Pada gambar 4.4 menunjukkan confusion matrix dari hasil pengujian model pada Fold 1 dalam skema validasi silang. Confusion matrix ini menggambarkan performa klasifikasi model dalam membedakan lima kelas yang merepresentasikan nama-nama jalan, yaitu: jalan_golf_kayu_tangan, jalan_kayu_tangan, jalan_veteran, jalan_ijen, dan jalan_suhat. Sumbu vertikal menunjukkan label sebenarnya (actual), sedangkan sumbu horizontal menunjukkan prediksi dari model.

Secara umum, nilai diagonal pada confusion matrix (dari kiri atas ke kanan bawah) merepresentasikan jumlah prediksi yang benar untuk masing-masing kelas. Diasumsikan model berhasil mengklasifikasikan 44 data jalan_kayu_tangan

dengan benar, 37 data jalan_veteran, dan 28 data jalan_suhat, yang menunjukkan bahwa model cukup mampu mengenali data dari kelas-kelas tersebut. Namun, terdapat beberapa kesalahan klasifikasi yang cukup signifikan, terutama pada kelas jalan_golf_kayu_tangan yang hanya berhasil diprediksi benar sebanyak 21 kali, sementara 26 data dari kelas ini justru salah diprediksi sebagai jalan_ijen.

Selain itu, terlihat bahwa terdapat confusion atau kebingungan antar beberapa kelas. Diasumsikan banyak data dari jalan_suhat yang diprediksi sebagai jalan_kayu_tangan (12 kali), serta data dari jalan_ijen yang tersebar cukup merata ke kelas lain, menunjukkan bahwa fitur dari data pada beberapa kelas ini kemungkinan memiliki kemiripan atau tumpang tindih dalam ruang fitur.

Confusion matrix ini penting untuk menganalisis kelemahan spesifik dari model, terutama untuk mengetahui pada kelas mana model sering melakukan kesalahan. Dari informasi ini, dapat dilakukan evaluasi lanjutan seperti data *augmentation*, *rebalancing class*, atau pemilihan fitur yang lebih informatif untuk meningkatkan akurasi klasifikasi di fold berikutnya.



Gambar 4.5 Confusion Matrix Model Testing Fold 2

Pada gambar 4.5 menggambarkan confusion matrix untuk pengujian model pada Fold ke-2 dari proses validasi silang. Setiap baris menunjukkan label aktual dari data, sedangkan kolom merepresentasikan prediksi model. Terdapat lima kelas lokasi yaitu jalan_golf_kayu_tangan, jalan_kayu_tangan, jalan_veteran, jalan_ijen, dan jalan_suhat.

Pada confusion matrix ini, terlihat bahwa kelas jalan_kayu_tangan memiliki performa terbaik, dengan 55 data diklasifikasikan dengan benar, dan hanya sedikit kesalahan prediksi (Diasumsikan 6 data diprediksi sebagai jalan_suhat). Ini

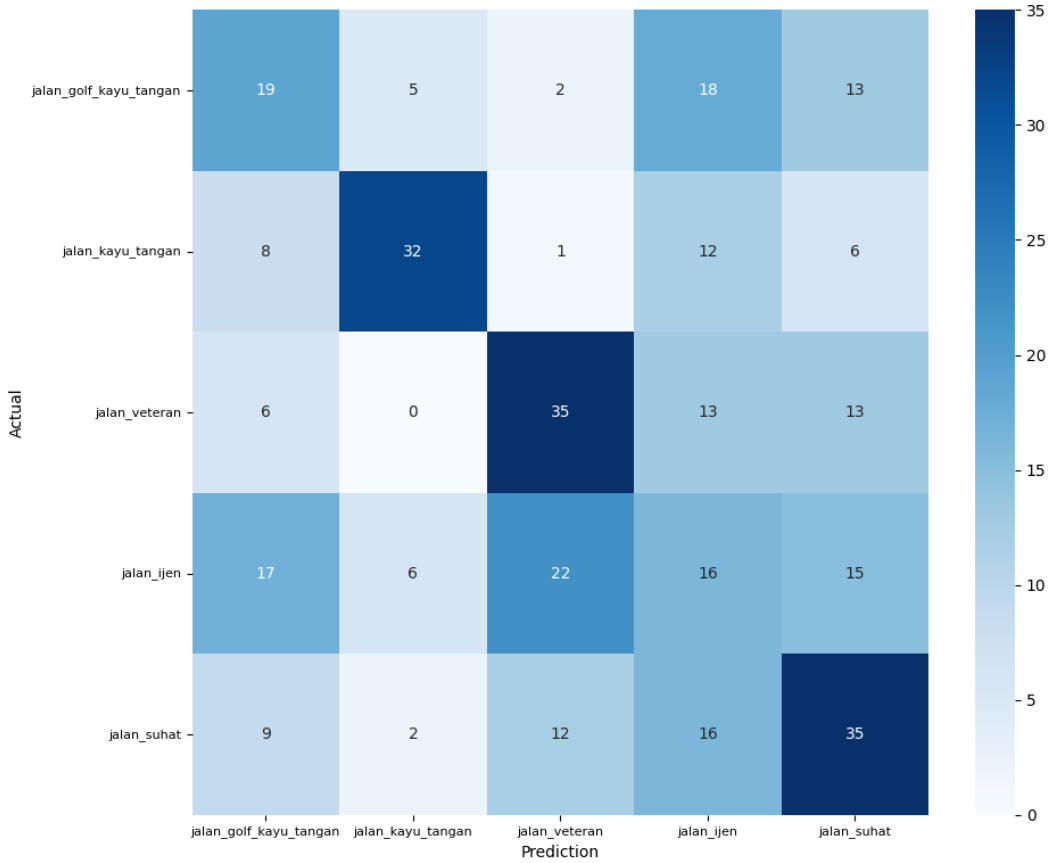
menunjukkan bahwa fitur-fitur yang dimiliki oleh data jalan_kayu_tangan cukup unik atau mudah dibedakan oleh model.

Namun, berbeda dengan fold sebelumnya, kelas jalan_veteran mengalami banyak kekeliruan klasifikasi. Meskipun 29 data berhasil diklasifikasikan dengan benar, terdapat 24 data yang salah diprediksi sebagai jalan_ijen, dan 13 data sebagai jalan_suhat. Hal ini mengindikasikan adanya kemiripan pola fitur antara jalan_veteran dan dua kelas tersebut, yang mungkin menyebabkan kebingungan pada model.

Kelas jalan_ijen juga menunjukkan sebaran prediksi yang cukup luas, dengan hanya 20 dari total data yang diprediksi dengan benar, dan sisanya tersebar ke kelas lain. Ini mengindikasikan bahwa model masih kesulitan mengenali pola khas dari jalan_ijen, yang mungkin disebabkan oleh data yang kurang representatif atau fitur yang kurang informatif. Sementara itu, kelas jalan_suhat kembali menunjukkan performa cukup baik, dengan 28 prediksi benar, walaupun masih ada kekeliruan seperti 12 data yang diklasifikasikan sebagai jalan_veteran.

Secara keseluruhan, confusion matrix Fold 2 menunjukkan bahwa meskipun beberapa kelas seperti jalan_kayu_tangan cukup stabil, model masih menunjukkan kebingungan dalam membedakan beberapa kelas lainnya, terutama

jalan_veteran dan jalan_ijen. Perbaikan pada fitur atau strategi klasifikasi mungkin diperlukan untuk meningkatkan akurasi keseluruhan model.



Gambar 4.6 Confusion Matrix Model Testing Fold 3

Pada gambar 4.6 confusion matrix pada Fold ke-3 ini menunjukkan distribusi hasil prediksi model terhadap lima kelas jalan. Sama seperti sebelumnya, sumbu vertikal menunjukkan label sebenarnya (actual), sedangkan sumbu horizontal menunjukkan hasil prediksi model (prediction). Tujuan utama dari visualisasi ini adalah untuk melihat seberapa baik model mengenali setiap kelas dan seberapa sering terjadi kesalahan klasifikasi antar kelas.

Pada Fold 3, kelas jalan_suhat menunjukkan performa yang paling konsisten, dengan 35 data berhasil diklasifikasikan dengan benar. Ini adalah jumlah

tertinggi dari seluruh kelas dalam fold ini, yang mengindikasikan bahwa fitur dari kelas jalan_suhat cukup kuat untuk dibedakan oleh model. jalan_veteran juga memiliki performa cukup baik, dengan 35 prediksi benar, walaupun masih ada 13 data yang salah diklasifikasikan ke jalan_ijen dan jalan_suhat.

Namun, dapat dilihat bahwa kelas jalan_ijen mengalami banyak kesalahan klasifikasi, hanya 16 data yang benar dari total, sementara sisanya tersebar ke berbagai kelas lainnya, terutama ke jalan_veteran (22 data) dan jalan_golf_kayu_tangan (17 data). Hal ini menunjukkan bahwa pola fitur untuk jalan_ijen mungkin tumpang tindih dengan kelas lain atau tidak cukup konsisten, menyebabkan model kesulitan membedakannya.

Sementara itu, jalan_kayu_tangan berhasil diprediksi dengan benar sebanyak 32 kali, tetapi masih terdapat cukup banyak kesalahan ke arah jalan_golf_kayu_tangan dan jalan_ijen. Ini menunjukkan bahwa meskipun performanya cukup baik, ada potensi perbaikan dengan menyempurnakan fitur atau representasi data.

Terakhir, jalan_golf_kayu_tangan memiliki 19 prediksi benar, namun juga mengalami penyebaran kesalahan ke beberapa kelas, terutama jalan_ijen (18 prediksi salah) dan jalan_suhat (13 prediksi salah). Ini mengindikasikan bahwa data dari kelas ini masih memiliki kesamaan pola dengan kelas-kelas lainnya yang mengganggu akurasi model.

Tabel 4.6 Detail Testing K Fold 3.

Fold	jalan_golf_kayu_tgn	jalan_kayu_tangan	jalan_veteran	jalan_ijen	jalan_suhat
1	21	44	37	25	28
2	19	55	29	20	28
3	19	32	35	16	35

Pada tabel 4.6 menyajikan hasil testing model klasifikasi pada masing-masing fold dari metode K-Fold Cross Validation, khususnya Fold ke-1 hingga Fold ke-3. Masing-masing kolom merepresentasikan jumlah prediksi yang benar (true positive) untuk setiap kelas jalan, seperti jalan_golf_kayu_tgn, jalan_kayu_tangan, jalan_veteran, jalan_ijen, dan jalan_suhat.

Secara umum, terlihat bahwa kelas jalan_kayu_tangan memiliki performa paling konsisten dan tinggi di setiap fold, dengan 44, 55, dan 32 prediksi benar berturut-turut. Ini menunjukkan bahwa model mampu mengenali ciri-ciri dari kelas ini dengan baik, kemungkinan karena karakteristik atau fitur visual/spasial yang lebih spesifik dibandingkan kelas lainnya.

Sementara itu, kelas jalan_suhat juga menunjukkan performa yang stabil, dengan jumlah prediksi benar yang cukup tinggi di setiap fold (28, 28, dan 35). Hal ini menunjukkan bahwa data untuk kelas ini memiliki konsistensi tinggi dan fitur-fitur yang cukup membedakannya dari kelas lainnya.

Sebaliknya, kelas jalan_ijen terlihat paling bermasalah, dengan penurunan akurasi signifikan terutama pada Fold ke-3, di mana hanya 16 data yang berhasil diklasifikasikan dengan benar. Penurunan ini selaras dengan hasil confusion matrix pada Fold 3 yang menunjukkan banyaknya data jalan_ijen yang salah diklasifikasikan ke kelas lain seperti jalan_golf_kayu_tangan dan jalan_veteran.

Untuk kelas jalan_veteran, jumlah prediksi benar cukup fluktuatif—dari 37 (Fold 1), menurun ke 29 (Fold 2), lalu naik ke 35 (Fold 3). Ini menunjukkan bahwa model memiliki performa yang cukup baik namun belum stabil untuk kelas ini, yang bisa jadi dipengaruhi oleh variasi data antar fold.

Selanjutnya, kelas jalan_golf_kayu_tgn memiliki performa cukup stabil tapi sedikit lebih rendah dibandingkan kelas lain, dengan jumlah prediksi benar sekitar 19–21. Hal ini mungkin disebabkan oleh tumpang tindih fitur dengan kelas seperti jalan_ijen atau jalan_suhat, yang bisa membingungkan model dalam melakukan prediksi.

Tabel 4.7 Detail Testing K Fold 5.

Fold	jalan_golf_kayu_tgn	jalan_kayu_tangan	jalan_veteran	jalan_ijen	jalan_suhat
1	15	28	25	7	20
2	19	28	17	17	12
3	14	28	13	9	18
4	13	21	22	15	20
5	9	21	22	19	20

Pada tabel 4.7 menunjukkan jumlah prediksi yang benar (true positive) pada masing-masing kelas untuk setiap fold dalam proses 5-Fold Cross Validation. Lima kelas yang diuji adalah jalan_golf_kayu_tgn, jalan_kayu_tangan, jalan_veteran, jalan_ijen, dan jalan_suhat. Hasil pada tabel ini merepresentasikan akurasi per kelas dalam tiap fold.

Tabel 4.8 Eval K 3.

Kelas	Precission	Recall	F1 Score
Jalan_golf_kayu_tangan	0.38	0.29	0.43
Jalan_kayu_tangan	0.62	0.60	0.60
Jalan_veteran	0.60	0.58	0.59
Jalan_ijen	0.28	0.44	0.34
Jalan_suhat	0.51	0.41	0.46

Berdasarkan hasil evaluasi performa model klasifikasi yang ditampilkan dalam tabel 4.8, terlihat bahwa kemampuan model dalam membedakan lokasi jalan berbeda-beda tergantung pada jumlah kelas yang digunakan dan karakteristik masing-masing kelas. Pada model dengan tiga kelas (k=3), performa terhadap kelas

"Jalan_golf_kayu_tangan" menunjukkan nilai precision sebesar 0.38, recall 0.29, dan F1 Score 0.43 dari total 72 data (support). Ini mengindikasikan bahwa model kurang mampu mengenali kelas ini dengan baik, ditandai dengan rendahnya akurasi prediksi maupun tingkat keberhasilan dalam menangkap data yang benar-benar termasuk ke dalam kelas tersebut.

Sementara itu, pada model dengan jumlah kelas yang lebih banyak, performa klasifikasi cenderung lebih stabil untuk beberapa kelas. Diasumsikan kelas "Jalan_kayu_tangan" dan "Jalan_veteran" masing-masing memiliki F1 Score sebesar 0.60 dan 0.59, yang menunjukkan keseimbangan antara precision dan recall yang cukup baik. Namun, masih terdapat kelas dengan performa rendah, seperti "Jalan_ijen" yang hanya memiliki precision 0.28 dan F1 Score 0.34, serta "Jalan_suhat" yang memiliki F1 Score 0.46. Hal ini menunjukkan bahwa meskipun model bisa mengenali beberapa kelas dengan baik, masih terdapat tantangan dalam membedakan kelas lain secara akurat.

Secara keseluruhan, performa model pada jumlah kelas yang lebih banyak terlihat sedikit lebih baik dibandingkan pada model dengan tiga kelas, khususnya untuk kelas-kelas tertentu. Namun, ketidakseimbangan antar kelas dalam hal representasi data dan fitur visual kemungkinan menjadi faktor penyebab utama rendahnya akurasi pada beberapa kelas. Oleh karena itu, peningkatan performa bisa dilakukan dengan pendekatan seperti augmentasi data, penyempurnaan fitur, atau pemilihan model yang lebih sesuai.

Tabel 4.9 Eval K 5.

Kelas	Precission	Recall	F1 Score
Jalan_golf_kayu_tangan	0.52	0.37	0.43
Jalan_kayu_tangan	0.62	0.65	0.54
Jalan_veteran	0.60	0.60	0.60
Jalan_ijen	0.19	0.23	0.21
Jalan_suhat	0.43	0.47	0.44

Berdasarkan tabel 4.9, merupakam evaluasi performa model klasifikasi dengan jumlah kelas lima (k=5), terlihat bahwa performa model bervariasi antar kelas. Kelas "Jalan veteran" menunjukkan performa terbaik dengan F1 Score sebesar 0.60, yang mencerminkan keseimbangan antara *precision* (0.60) dan *recall* (0.60). Disusul oleh kelas "Jalan kayu tangan" dengan F1-*Score* 0.54 dan *precision* yang cukup tinggi yaitu 0.62. Hal ini menunjukkan bahwa model cukup mampu mengenali gambar dari dua kelas tersebut secara konsisten dan akurat.

Sementara itu, kelas "Jalan golf kayu tangan" memiliki F1 Score yang lebih rendah, yaitu 0.43, dengan *precision* 0.52 dan *recall* 0.37. Ini menunjukkan bahwa meskipun model cukup sering mengklasifikasikan gambar ke kelas ini, banyak di antaranya tidak benar-benar berasal dari kelas tersebut (*recall* rendah). Hal yang serupa terjadi pada kelas "Jalan suhat" yang memiliki F1 Score 0.44, menandakan bahwa model belum cukup baik dalam mengenali karakteristik visual dari kelas ini secara menyeluruh.

Yang paling menonjol dari hasil evaluasi ini adalah performa model terhadap kelas "Jalan ijen", yang memiliki nilai precision dan recall sangat rendah (0.19 dan 0.23) serta F1 Score hanya sebesar 0.21. Ini menandakan bahwa model kesulitan membedakan gambar-gambar yang termasuk ke dalam kelas ini, kemungkinan karena ciri visualnya kurang khas atau data pelatihan tidak cukup representatif.

Kesimpulannya, meskipun model mampu mengklasifikasikan beberapa kelas dengan cukup baik, masih diperlukan peningkatan terutama pada kelas dengan performa rendah, diasumsikan melalui penyeimbangan jumlah data, peningkatan kualitas fitur visual, atau tuning model yang lebih spesifik.

Secara umum, kelas jalan_kayu_tangan kembali menjadi yang paling konsisten dan dominan, dengan prediksi benar yang stabil pada angka 28 di tiga fold pertama, dan sedikit menurun menjadi 21 pada dua fold terakhir. Hal ini menandakan bahwa fitur dari kelas ini cukup kuat dan mudah dikenali oleh model.

Kelas jalan_golf_kayu_tgn dan jalan_veteran menunjukkan fluktuasi kinerja. Pada jalan_golf_kayu_tgn, terlihat adanya tren penurunan dari Fold 1 (15) hingga Fold 5 (9). Penurunan ini bisa disebabkan oleh berkurangnya representasi data yang khas dari kelas ini di fold tersebut. Sedangkan jalan_veteran mulai dari 25 di Fold 1, turun drastis di Fold 3 (13), lalu kembali naik di Fold 4 dan 5, menunjukkan ketidakstabilan model dalam mengenali ciri-ciri dari kelas ini secara konsisten.

Untuk kelas jalan_ijen, performa model cukup bervariasi, mulai dari 7 prediksi benar di Fold 1 hingga 19 di Fold 5. Kenaikan jumlah prediksi benar di fold terakhir ini bisa menandakan bahwa data pada Fold 5 lebih representatif atau model telah belajar lebih baik untuk mengenali pola dari kelas ini seiring proses pelatihan antar fold.

Kelas jalan_suhat menunjukkan performa yang cukup stabil, dengan hasil yang berkisar antara 12 hingga 20 prediksi benar. Hal ini menunjukkan bahwa

meskipun terdapat sedikit variasi antar fold, model cukup mampu mengenali kelas ini dengan baik dan relatif konsisten.

4.3 Pembahasan

Proses pelatihan model dengan menggunakan metode K-Fold Cross Validation memberikan gambaran menyeluruh mengenai performa model dalam mengklasifikasikan citra jalan berdasarkan beberapa kategori, yaitu jalan_golf_kayu_tgn, jalan_kayu_tangan, jalan_veteran, jalan_ijen, dan jalan_suhat. Berdasarkan Tabel 4.5, terlihat bahwa jumlah *epoch* yang dibutuhkan untuk mencapai konvergensi bervariasi pada tiap fold, mulai dari 21 hingga 54 *epoch*. Nilai loss terkecil tercapai pada fold ke-2 dan ke-5 (0.0966), sedangkan akurasi tertinggi terdapat pada fold pertama (0.475). Hal ini menunjukkan bahwa kemampuan model dalam belajar sangat dipengaruhi oleh distribusi data pada tiap fold.

Analisis lebih mendalam melalui confusion matrix pada Fold 1 hingga Fold 3 menunjukkan bahwa kelas jalan_kayu_tangan konsisten mendapatkan jumlah prediksi yang benar lebih tinggi dibanding kelas lainnya. Hal ini mengindikasikan bahwa fitur visual pada kelas ini paling dapat dibedakan oleh model. Sebaliknya, kelas seperti jalan_ijen dan jalan_golf_kayu_tgn memiliki tingkat kesalahan klasifikasi yang cukup tinggi, terbukti dari banyaknya prediksi yang tersebar ke kelas lain. Jjalan_ijen kerap diklasifikasikan sebagai jalan_golf_kayu_tgn dan jalan_veteran, yang mengindikasikan adanya kemiripan fitur antar kelas tersebut.

Dalam Tabel 4.7 dan Tabel 4.8, dapat dilihat hasil prediksi benar pada masing-masing kelas per fold. Data ini memperkuat temuan sebelumnya bahwa jalan_kayu_tangan merupakan kelas yang paling mudah dikenali, dengan nilai prediksi benar tertinggi secara konsisten. Selain itu, jalan_suhat juga menunjukkan performa yang stabil, terutama pada 3-Fold. Namun, kelas seperti jalan_ijen dan jalan_golf_kayu_tgn terlihat lebih fluktuatif, yang menunjukkan bahwa model kesulitan dalam mengenali karakteristik spesifik dari kelas-kelas tersebut secara konsisten di berbagai fold.

Secara keseluruhan, hasil evaluasi ini menunjukkan bahwa model memiliki performa klasifikasi yang cukup baik pada beberapa kelas, namun masih memerlukan perbaikan dalam membedakan kelas yang memiliki fitur visual yang mirip. Beberapa upaya yang dapat dilakukan untuk meningkatkan performa antara lain adalah penambahan data pelatihan, augmentasi data untuk kelas yang sulit dikenali, serta eksplorasi arsitektur model yang lebih kompleks atau teknik regularisasi yang lebih baik. Evaluasi model menggunakan K-Fold juga terbukti penting untuk mendapatkan gambaran performa yang lebih stabil dan menghindari overfitting terhadap data tertentu.

Dalam Islam, ilmu dan teknologi adalah anugerah yang harus dimanfaatkan untuk kebaikan umat manusia. Hubungan manusia dengan Allah (hablum minallah) dan hubungan antar sesama manusia (hablum minannas) menjadi dasar utama dalam setiap aktivitas, termasuk penelitian. Penelitian ini tidak hanya bertujuan untuk meningkatkan akurasi model, tetapi juga diharapkan dapat memberikan manfaat yang lebih luas bagi masyarakat dalam hal pemetaan lokasi dan

pengelolaan kota yang lebih efektif.tercermin dalam implementasi prinsip-prinsip Islam sepanjang proses penelitian.

Selain itu, hubungan antar manusia juga dijaga dengan menghindari tindakan yang merugikan orang lain. Firman Allah SWT dalam Q.S. Al-Maidah: 2 menegaskan pentingnya tolong-menolong dalam kebaikan:

وَلَآ آٰمِّيْنَ الْبَيْتَ الْحَرَامَ يَبْتَغُوْنَ فَضْلًا مِّنْ رَّبِّهِمْ وَرِضْوَانًاۗ وَاِذَا حَلَلْتُمْ فَاصْطَادُوْاۗ وَلَا يَجْرِمَنَّكُمْ شَنَاٰنُ قَوْمٍ اَنْ صَدُّوْكُمْ عَنِ الْمَسْجِدِ الْحَرَامِ اَنْ تَعْتَدُوْاۘ وَتَعَاوَنُوْا عَلَى الْبِرِّ وَالتَّقْوٰىۖ وَلَا تَعَاوَنُوْا عَلَى الْاِثْمِ وَالْعُدْوَانِۖ وَاتَّقُوا اللّٰهَۗ اِنَّ اللّٰهَ شَدِيْدُ الْعِقَابِ

"Dan tolong-menolonglah kamu dalam (mengerjakan) kebajikan dan takwa, dan jangan tolong-menolong dalam berbuat dosa dan pelanggaran"(QS.Al-Maidah[5:2]).

Dalam penelitian ini, hasil yang dihasilkan tidak hanya berorientasi pada pencapaian teknis, tetapi juga memberikan solusi yang sesuai dengan kebutuhan masyarakat. Teknologi yang dikembangkan harus mencerminkan nilai-nilai Islam, seperti transparansi, kejujuran, dan tidak mengandung unsur yang bertentangan dengan syariat, seperti gharar (ketidakjelasan) atau penipuan.

مَن فَرَّجَ عَنْ مُسْلِمٍ كُرْبَةً مِنْ كُرُوبِ الدُّنْيَا، فَرَّجَ اللَّهُ عَنْهُ كُرْبَةً مِنْ كُرُوبِ يَوْمِ الْقِيَامَةِ، وَاللَّهُ فِي عَوْنِ الْعَبْدِ مَا دَامَ الْعَبْدُ فِي عَوْنِ أَخِيهِ.

(رواه مسلم)

"Barang siapa melepaskan dari seorang muslim suatu kesulitan di dunia, Allah akan melepaskan kesulitan darinya pada hari kiamat; dan Allah senantiasa menolong hamba-Nya selama ia (suka) menolong saudaranya" (HR. Abu Hurairah dan Muslim).

Hadis yang diriwayatkan Muslim dari Abu Hurairah RA ini menjelaskan bahwa siapa saja yang menghilangkan kesusahan seorang muslim di dunia, maka Allah akan menghilangkan kesusahannya pada hari kiamat. barang siapa

memudahkan urusan orang yang kesulitan, Allah akan memudahkan urusannya di dunia dan akhirat; dan siapa yang menutupi aib saudaranya, maka Allah akan menutupi aibnya di dunia dan akhirat. Imam Nawawi dalam Syarh Shahih Muslim menjelaskan bahwa "menghilangkan kesusahan" berarti benar-benar membantu menyelesaikan beban saudaranya, sedangkan "memudahkan orang susah" bisa dilakukan dengan menunda tagihan atau memberi bantuan finansial secara langsung. semuanya akan dibalas Allah dengan kemudahan pula. Beliau juga menafsirkan bahwa menutup aib berarti tidak menyebarkan kesalahan yang diketahui dari orang lain, dan sebagai balasan, Allah akan menjaga kehormatan orang itu di dunia dan akhirat.

Terakhir, makna "Allah menolong hamba selama ia menolong saudaranya" menunjukkan bahwa siapa pun yang tulus membantu kebutuhan orang lain, maka pertolongan Allah akan menyertainya secara langsung karena balasan dari Allah selalu sejenis dengan amal yang dilakukan (Tafsir Imam Nawawi).

BAB V

KESIMPULAN DAN SARAN

Berdasarkan hasil evaluasi menggunakan metode K-Fold Cross Validation baik dalam 5-Fold maupun 10-Fold, dapat disimpulkan bahwa model klasifikasi yang dibangun mampu mengenali beberapa kelas citra jalan dengan cukup baik, khususnya pada kelas jalan_kayu_tangan dan jalan_suhat. Model menunjukkan kestabilan performa pada beberapa fold, namun juga ditemukan variasi akurasi antar fold yang menunjukkan adanya pengaruh signifikan dari distribusi data pelatihan dan pengujian. Nilai loss yang rendah pada beberapa fold juga mengindikasikan bahwa model mampu belajar dengan cukup baik dari data yang diberikan.

Hasil *confusion matrix* mengungkapkan bahwa kelas-kelas tertentu seperti jalan_ijen dan jalan_golf_kayu_tgn masih sering mengalami salah klasifikasi. Hal ini bisa disebabkan oleh kemiripan visual antar kelas atau kurangnya representasi data yang memadai pada kelas-kelas tersebut. Meskipun demikian, secara umum model mampu memberikan prediksi yang cukup akurat dengan proporsi klasifikasi yang benar relatif tinggi, terutama ketika data terdistribusi lebih merata.

Untuk meningkatkan performa model di masa mendatang, beberapa langkah strategis dapat dipertimbangkan. Pertama, perlu dilakukan augmentasi data untuk menyeimbangkan jumlah citra antar kelas, sehingga model dapat belajar representasi yang lebih baik dan tidak bias terhadap kelas mayoritas. Kedua, eksplorasi terhadap arsitektur model yang lebih kompleks seperti CNN yang lebih

dalam, atau penggunaan pretrained model seperti ResNet atau MobileNet, berpotensi meningkatkan akurasi secara keseluruhan.

Selain itu, disarankan untuk melakukan analisis fitur lebih lanjut untuk memahami karakteristik visual tiap kelas, sehingga preprocessing atau teknik segmentasi dapat disesuaikan. Evaluasi model sebaiknya juga tidak hanya fokus pada akurasi, tetapi memperhatikan metrik lain seperti *precision*, *recall*, dan F1-*score* agar gambaran performa lebih menyeluruh. Terakhir, perlu dilakukan pengujian dalam skenario nyata atau *real-time* untuk melihat bagaimana model bekerja pada kondisi yang lebih bervariasi dan dinamis.

Dilihat dari hasil rata-rata uji skenario untuk K3 dan K5 yang memiliki nilai akurasi rata-rata sebesar 0.453 dan 0.443, dapat disimpulkan bahwa K3 menunjukkan performa yang lebih optimal dibandingkan dengan K5. Perbedaan nilai ini menunjukkan bahwa parameter K3 lebih sesuai digunakan dalam konteks pengujian yang dilakukan.

Dengan demikian, model berhasil mencapai akurasi yang paling ideal berdasarkan skenario-skenario pengujian yang telah dilakukan. Hasil ini memberikan indikasi kuat bahwa K3 dapat dijadikan pilihan utama dalam konfigurasi model untuk mencapai hasil terbaik.

DAFTAR PUSTAKA

Ali, M. M., Faishol M., Anwar K., (2022). Deteksi Jalan Berlubang Menggunakan Metode Grey Level Co-Occurrence Matrix Dan Neural Network. *Jurnal Kecerdasan Buatan, Komputasi dan Teknologi Informasi.* Vol. 3 No.1 Tahun 2022. https://ejournal.unuja.ac.id/index.php/core. E-ISSN: 2774-7875 and P-ISSN: 2775-0124

Amarullah, M. Fiqri (2024) Analisis Prediksi Tutupan Lahan Kota Malang Berbasis Cellular Automata Terhadap Data Rtrw Tahun 2022-2042. *Skripsi thesis, ITN Malang.*

Azzah , Roudhoh (2023) High Performance Computing Untuk Klasifikasi Image Tumbuhan Obat Sirih Dan Binahong Menggunakan Metode Convolutional Neural Network (CNN). *Matematika Dan Ilmu Pengetahuan Alam*, Universitas Lampung.

Devi, Muhammad Resa, Arif Yudianto, Pristi Sukmasetya. 2024. Klasifikasi Citra Candi Berdasarkan Tekstur Bentuk Menggunakan Convolutional Neural Network. *Jurnal Informatika Polinema. JIP (Jurnal Informatika Polinema)* ISSN: 2614-6371 E-ISSN: 2407-070X

E. Setiadi, A. Wibowo. (2024). Klasifikasi dan Deteksi Keretakan Pada Trotoar Menggunakan Metode Convolutional Neural Network.

Fathur Rizal, Fuadz Hasyim, Kamil Malik, Yudistira Yudistira. 2022. Implementasi Algoritma Convolutional Neural Networks (CNN) Untuk Klasifikasi Batik. *COREAI: Jurnal Kecerdasan Buatan, Komputasi dan Teknologi Informasi*. https://ejournal.unuja.ac.id/index.php/core. E-ISSN : 2774-7875 and P-ISSN : 2775-0124

Fernando, Muhammad Ativ, Dyah Ayu. 2020. Penerapan Algoritma A-Star Pada Aplikasi Pencarian Lokasi Fotografi Di Bandar Lampung Berbasis Android. *Jurnal Teknoinfo*, Vol. 14, No. 1, 2020, 27-34, ISSN: 2615-224XDOI: 10.33365/jti.v14i1.509

Ghassani, Shabrina Amalia. 2022. Identifikasi Potensi Gastronomy Tourism Di Kota Malang. Warta Pariwisata. ISSN 2773-4723 [online] ISSN 1410-7112 [print] https://journals.itb.ac.id/index.php/wpar. http://doi.org/10.5614/wpar.2022.20.2.03

Hapsary, Maharany Shandra Ayu. (2021). Analisis Prediksi Perubahan Penggunaan Lahan Dengan Pendekatan Artificial Neural Network Dan Regresi Logistik Di Kota Balikpapan. *Jurnal Geodesi Undip.* Volume [10] , Nomor [2] , Tahun 2021, (ISSN : 2337-845X)

Islamweb. (n.d.). *Bab Keutamaan Berkumpul untuk Membaca Al-Qur'an dan Berdzikir*. Diakses pada 18 Mei 2025, dari https://www.islamweb.net/ar/library/content/53/7883/باب-فضل-الاجتماع-على-تلاوة-القرآن-وعلى-الذكر#:~:text=20%20%20حديث20%فيهnindex,20%العلم20%طلب20%في20%المشي،

Khairul Azmi, Sarjon Defit, S. Sumijan. 2023. Implementasi Convolutional Neural Network (CNN) Untuk Klasifikasi Batik Tanah Liat Sumatera Barat. *Jurnal UNITEK*. Vol.16 No.1 Januari - Juni 2023 e-ISSN : 2580-2582, p-ISSN: 2089-3957.

Muhammad Gusanwa. (2023). Implementasi Neural Network Untuk Klasifikasi Fungsi Barang Milik Daerah Pemerintah Kota Madiun Berbasis Web

Putri, Safitri A. 2023. Analisis User Experience Aplikasi Wisata Kota Malang Untuk Wisatawan Lokal. *Journal of Social Economics Research*. Volume 5, Issue 2, December 2023P-ISSN: 2715-6117E-ISSN: 2715-6966Open Access at: https://idm.or.id/JSER/index.php/JSER

Radikto, Mulyana, I. D., Rofik, A. M., Zakaria, Z. O. (2022). Klasifikasi Kendaraan pada Jalan Raya menggunakan Algoritma Convolutional Neural Network (CNN). *STIKOM CKI Sekolah Tinggi Ilmu Komputer Cipta Karya Informatika*. Halaman 1668-1679. Volume 6 Nomor 1 Tahun 2022. ISSN: 2614-6754 (print) ISSN: 2614-3097(online)

Ricky Adytya Pratama, Sentot Achmadi, Karina Auliasari. 2022. Penerapan Metode Convolutional Neural Network Pada Aplikasi Deteksi Wajah Pengunjung Perpustakaan. *JATI (Jurnal Mahasiswa Teknik Informatika*) Vol. 6 No. 1, Februari 2022

Subarkah, Muhammad Zidni, H.Martina, E. Zukhronah. 2022. Analisis Sentimen Review Tempat Wisata Pada Data Online Travel Agency Di Yogyakarta Menggunakan Model Neural Network IndoBERTweet Fine Tuning. *Seminar Nasional Official Statistics* 2022.

Sumartha, N. N. C., Wijaya I. G. P, Bimantoro, F., Nugraha, G. S., (2023). Klasifikasi Citra Lubang Pada Permukaan Jalan Beraspal dengan Metode

Convolutional Neural Networks. *J-COSINE (Journal of Computer Science and Informatics Engineering)*. Vol. 8, No. 1, June 2024. Accredited Sinta-4 by RISTEKDIKTI Decree No. 79/E/KPT/2023 E-ISSN:2541-0806 P-ISSN:2540-8895

TafsirWeb. (2023, 25 Oktober). *Tafsir Surat Al-Ankabut Ayat 69*. Diakses pada 31 Oktober 2023, dari https://tafsirweb.com/7295-surat-al-ankabut-ayat-69.html

Yusup Yulianto, Ari Wibowo. (2023). Deteksi Keretakan Jalan Aspal Menggunakan Metode Convolutional Neural Network. *Jurnal Teknik Sipil Cendekia.* Vol 4 No 2 (2023), July 2023, pp. 581-594

LAMPIRAN

Lampiran 1: Perhitungan Manual

Berikut merupakan contoh perhitungan dimulai dari input pixel sampai output menggunakan pixel dari image berikut sebagai contoh input.

Gambar 3. 6 Potongan Jalan Golf Kayu Tangan 60 x 60

Pada pixel pertama dari gambar 3.5 adalah (10, 63, 43), disini kita akan menggunakan nilai pertama yaitu red dengan nilai 10. x = 10, bobot w = 0.5, dan bias b = 2, maka perhitunganya sebagai berikut.

$$w = (10\ x\ 0.5) + 2$$

$$w = 5 + 2 = 7$$

Maka hasil dari setelah aktivasi ReLU adalah:

$$h_i = \max(0, 7) = 7$$

Hasil dari aktivasi ReLU sebelumnya adalah 7, bobot v = 0.5, dan bias b_2 = 2, maka:

$$v_{\mathrm{i}} = (7\ x\ 0.5) + 2$$

$$v_{\mathrm{i}} = 3.5 + 2 = 5.5$$

Diasumsikan hasil dari hidden layer 2 untuk ke 5 class yang ada sebagai berikut, h_1 = 5.5, h_2 = 15.0, h_3 = 10.0, h_4 = 5.0, h_5 = 1.0, dilanjutkan menghitung eksponensial dari setiap nilai:

$$\exp(5.5) = 244.69$$

$$\exp(15.0) = 3.27\ x\ 10^6$$

$$\exp(10.0) = 2.20\ x\ 10^4$$

$$\exp(5.0) = 148.41$$

$$\exp(1.0) = 2.72$$

Kemudian dilanjutkan untuk menghitung total eksponensial

$$S = (244.69) + (3.27\ x\ 10^6) + (2.20\ x\ 10^4) + (148.41) + (2.72)$$

$$S = 3.29\ x\ 10^6$$

Lalu hitung probabilitas untuk setiap kelas

$$\hat{o}_1 = \frac{244.69}{3.29\ x\ 10^6} = 7.43\ x\ 10^{-5}$$

$$\hat{o}_2 = \frac{3.27\ x\ 10^6}{3.29\ x\ 10^6} = 0.99$$

$$\hat{o}_3 = \frac{2.20\ x\ 10^4}{3.29\ x\ 10^6} = 0.0067$$

$$\hat{o}_4 = \frac{148.41}{3.29\ x\ 10^6} = 4.51\ x\ 10^{-5}$$

$$\hat{o}_5 = \frac{2.72}{3.29\ x\ 10^6} = 8.27\ x\ 10^{-9}$$

Pada hasil dari perhitungan tersebut, bisa di lihat bahwa hasil tertinggi terdapat pada output ke 2, yang mengindikasikan tinkat keyakinan terhadap *class* tersebut adalah 99%.

Lampiran 2: Code Program

```
# ------------------------------------------------------------------------
-------
```

```
# Fungsi aktivasi dan derivatifnya
def relu(x):
    return np.maximum(0, x)

def relu_derivative(x):
    return np.where(x > 0, 1, 0)

def softmax(x):
    e_x = np.exp(x - np.max(x, axis=-1, keepdims=True))
    return e_x / np.sum(e_x, axis=-1, keepdims=True)

def safe_relu(x):
    result = relu(x)
    assert not np.isnan(result).any(), "Ada nilai NaN setelah ReLU"
    return result

def safe_softmax(x):
    result = softmax(x)
    assert not np.isnan(result).any(), "Ada nilai NaN setelah Softmax"
    return result

# ---------------------------------------------------------------------
-------
# Fungsi untuk meng-update bobot
def update_weights(weights, biases, dweights, dbiases, learning_rate):
    weightsUpdate = weights - learning_rate * dweights
    biasesUpdate = biases - learning_rate * dbiases

    assert  not  np.isnan(weightsUpdate).any(),  "Ada  nilai  NaN  dalam
pembaruan bobot"
    assert  not  np.isnan(biasesUpdate).any(),  "Ada  nilai  NaN  dalam
pembaruan bias"

    # Check for NaN values
    if np.isnan(weightsUpdate).any() or np.isnan(biasesUpdate).any():
        # raise ValueError("Encountered NaN values in weights or biases")
        print("NaN values encountered in weight or bias updates.")
        return weights, biases

    return weightsUpdate, biasesUpdate

# ---------------------------------------------------------------------
-------
# Fungsi untuk membaca data dari folder
def load_data(file_path):
    df = pd.read_excel(file_path)
    images = df.iloc[:, :-1].values  # Ambil semua kolom kecuali kolom
terakhir (label)
    labels = df.iloc[:, -1].values  # Ambil kolom label
    return images, labels

# ---------------------------------------------------------------------
-------
```

```
# Proses read data dan melakukan training
def   train_model(data,   labels,   learning_rate=0.01,   name_model="",
min_loss=0.1, tolerance=0.0001):
    assert not np.isnan(data).any(), "Data mengandung NaN"
    assert not np.isnan(labels).any(), "Labels mengandung NaN"

    print('prepare data')
    np.random.seed(42)  # Tetapkan seed agar hasil dapat direproduksi

    # Inisialisasi bobot dan bias
    input_size = data.shape[1]
    output_size = len(np.unique(labels))

    class_names    =    ['jalan_golf_kayu_tangan',    'jalan_kayu_tangan',
'jalan_veteran', 'jalan_ijen', 'jalan_suhat']
    all_predictions = []  # Untuk menyimpan semua prediksi
    all_labels = []       # Untuk menyimpan semua label sebenarnya
    losses = []           # Menyimpan nilai loss setiap epoch

    hidden_weights  =  np.random.randn(input_size,  60)  *  np.sqrt(2.  /
input_size)
    hidden_biases = np.zeros(60)
    output_weights = np.random.randn(60, output_size) * np.sqrt(2. / 60)
    output_biases = np.zeros(output_size)

    previous_loss  =  float('inf')   #  Untuk  menyimpan  loss  dari  epoch
sebelumnya
    epoch = 0
    print('after prepare')

    while True:  # Loop tanpa batas hingga kondisi berhenti terpenuhi
        total_loss = 0.0
        correct_predictions = 0
        epoch_predictions = []  # Prediksi pada epoch ini
        epoch_labels = []       # Label pada epoch ini

        for i in range(len(data)):
            image = data[i]
            label = labels[i]

            # Forward pass
            hidden_input = np.dot(image, hidden_weights) + hidden_biases
            hidden_output = safe_relu(hidden_input)
            final_input   =   np.dot(hidden_output,   output_weights)   +
output_biases
            final_output = safe_softmax(final_input)

            # Hitung loss
            epsilon = 1e-10
            loss = -np.log(final_output[label] + epsilon)

            # Akumulasi total loss
            total_loss += loss
```

```
            # Prediksi dan akurasi
            predicted_label = np.argmax(final_output)
            if predicted_label == label:
                correct_predictions += 1

            # Simpan prediksi dan label pada epoch ini
            epoch_predictions.append(predicted_label)
            epoch_labels.append(label)

            # Backpropagation
            error = np.zeros_like(final_output)
            error[label] = final_output[label] - 1
            doutput_weights    =    np.dot(hidden_output.reshape(-1,    1),
error.reshape(1, -1))
            doutput_biases = error
            dhidden      =      np.dot(output_weights,      error)      *
relu_derivative(hidden_input)
            dhidden_weights      =      np.dot(image.reshape(-1,      1),
dhidden.reshape(1, -1))
            dhidden_biases = dhidden

            # Update bobot dan bias
            output_weights,                output_biases                =
update_weights(output_weights,      output_biases,      doutput_weights,
doutput_biases, learning_rate)
            hidden_weights,                hidden_biases                =
update_weights(hidden_weights,      hidden_biases,      dhidden_weights,
dhidden_biases, learning_rate)

        # Cetak loss dan akurasi setiap epoch
        epoch += 1
        avg_loss = total_loss / len(data)
        accuracy = correct_predictions / len(data)
        print(f'Epoch   {epoch}   -   Loss:   {avg_loss:.4f},   Accuracy:
{accuracy:.4f}')

        # Simpan loss ke daftar
        losses.append(avg_loss)

        # Simpan prediksi dan label hanya pada epoch terakhir
        all_predictions = epoch_predictions
        all_labels = epoch_labels

        # Kondisi berhenti jika loss sudah mencapai min_loss atau perubahan
loss < tolerance
        if avg_loss < min_loss:
            print(f"Training  stoped  at  epoch  {epoch}  because  loss  <
{min_loss:.4f}")
            break
        elif abs(previous_loss - avg_loss) < tolerance:
            print(f"Training stoped at epoch {epoch} because changes loss
< {tolerance:.4f}")
```

```
            break

        previous_loss = avg_loss

    print("Confusion Matrix Training")
    # Cetak confusion matrix
    plot_confusion_matrix(all_labels,    all_predictions,    class_names,
name_model)

    print("")
    print("Loss Convergence Training")
    # Tampilkan grafik loss
    plot_loss_curve(losses, title=f"Loss Convergence {name_model}")

    return hidden_weights, hidden_biases, output_weights, output_biases,
losses

def plot_confusion_matrix(labels, predictions, class_names, name_model):
    """
    Fungsi untuk membuat dan menampilkan confusion matrix sebagai heatmap.
    Args:
        labels (array-like): Label aktual.
        predictions (array-like): Prediksi model.
        class_names (list): Daftar nama kelas.
    """
    cm          =          confusion_matrix(labels,          predictions,
labels=range(len(class_names)))

    plt.figure(figsize=(10, 8))
    sns.heatmap(cm,       annot=True,       fmt="d",       cmap="Blues",
xticklabels=class_names, yticklabels=class_names, cbar=True)

    plt.title("Confusion Matrix Model " + name_model, fontsize=13)
    plt.xlabel("Prediction", fontsize=10)
    plt.ylabel("Actual", fontsize=10)
    plt.xticks(fontsize=8, rotation=0)
    plt.yticks(fontsize=8, rotation=0)
    plt.tight_layout()
    plt.show()

def plot_loss_curve(losses, title="Loss Convergence"):
    """
    Fungsi untuk mencetak grafik konvergensi loss selama training.

    Parameters:
    - losses: List of float, berisi nilai loss untuk setiap epoch.
    - title: str, judul grafik.

    Returns:
    - None
    """
    plt.figure(figsize=(10, 6))
```

```
    plt.plot(range(1, len(losses) + 1), losses, marker='o', color='b',
label='Loss')
    plt.title(title, fontsize=16)
    plt.xlabel("Epoch", fontsize=14)
    plt.ylabel("Loss", fontsize=14)
    plt.grid(True, linestyle='--', alpha=0.7)
    plt.legend(fontsize=12)
    plt.tight_layout()
    plt.show()

# ---------------------------------------------------------------------
-------
# Save model
def save_model(filename, model):
    with open(filename, 'wb') as file:
        pickle.dump(model, file)

# ---------------------------------------------------------------------
-------
# Load model
# Fungsi untuk membaca data dari file Excel
def load_data(file_path):
    df = pd.read_excel(file_path)
    images = df.iloc[:, :-1].values  # Ambil semua kolom kecuali kolom
terakhir (label)
    labels = df.iloc[:, -1].values  # Ambil kolom label

    # Konversi label teks menjadi angka
    label_to_int = {'jalan_golf_kayu_tangan': 0, 'jalan_kayu_tangan': 1,
'jalan_veteran': 2, 'jalan_ijen': 3, 'jalan_suhat': 4}
    # label_to_int = {'ruang_kelas': 0, 'ruang_lab_mobile': 1}
    labels = np.array([label_to_int[label] for label in labels])

    return images, labels

# ---------------------------------------------------------------------
-------
# Testing dengan memilih gambar yang ingin diklasifikasikan
def    classify_image(hidden_weights,    hidden_biases,    output_weights,
output_biases, image_path, threshold=0.05):
    # Lokasi file mean dan std
    mean_std_file_path                                                  =
"/content/drive/MyDrive/collection/datas/skripsi/mean_std_params.pkl"

    # Load mean dan std dari file
    try:
        with open(mean_std_file_path, "rb") as f:
            normalization_params = pickle.load(f)
    except FileNotFoundError:
        raise FileNotFoundError(f"File mean dan std tidak ditemukan di
lokasi: {mean_std_file_path}")

    # Ambil nilai mean dan std dari data training
```

```
    mean_r_training = normalization_params["mean_r"]
    std_r_training = normalization_params["std_r"]
    mean_g_training = normalization_params["mean_g"]
    std_g_training = normalization_params["std_g"]
    mean_b_training = normalization_params["mean_b"]
    std_b_training = normalization_params["std_b"]

    # Load dan proses gambar
    image_data_r,          image_data_g,          image_data_b          =
load_and_process_image(image_path)

    # Normalisasi gambar menggunakan mean dan std dari training
    image_data_r = (image_data_r - mean_r_training) / std_r_training
    image_data_g = (image_data_g - mean_g_training) / std_g_training
    image_data_b = (image_data_b - mean_b_training) / std_b_training

    # Gabungkan saluran warna menjadi satu vektor (RGB)
    image_data_concat   =   np.concatenate((image_data_r,   image_data_g,
image_data_b), axis=0)

    # Pastikan dimensi data gambar sesuai dengan model
    assert    image_data_concat.shape[0]    ==    hidden_weights.shape[0],
f"Dimensi  data  gambar  {image_data_concat.shape[0]}  tidak  sesuai  dengan
dimensi bobot {hidden_weights.shape[0]}."

    # Proses klasifikasi
    hidden_input    =    np.dot(image_data_concat,    hidden_weights)    +
hidden_biases
    hidden_output = safe_relu(hidden_input)
    final_input = np.dot(hidden_output, output_weights) + output_biases
    final_output = safe_softmax(final_input)

    # Penentuan label berdasarkan hasil akhir
    class_label = np.argmax(final_output)
    label_to_class = {0: 'jalan_golf_kayu_tangan', 1: 'jalan_kayu_tangan',
2: 'jalan_veteran', 3: 'jalan_ijen', 4: 'jalan_suhat'}
    original_label = label_to_class[class_label]

    confidence_level = final_output[class_label]
    print(f"Final Output: {final_output}")

    # Cek confidence level
    if np.isnan(confidence_level) or confidence_level < threshold:
        confidence_level = 0

    return final_output

# ----------------------------------------------------------------------
-------
# Pilih gambar dari file lokal untuk uji model
def choose_image():
    uploaded = files.upload()
    for file_name in uploaded.keys():
```

```
        return file_name

# Fungsi untuk memuat dan memproses gambar
def load_and_process_image(image_path):
    image = Image.open(image_path).convert('RGB')
    image = image.resize((60, 60))  # Ubah ukuran sesuai dengan input model
Anda (misal 60x60 untuk menghasilkan 3600 fitur)
    # image_data = np.array(image) / 255.0  # Normalisasi ke rentang 0-1
    image_data = np.array(image)
    image_data_r = image_data[:, :, 0].flatten()
    image_data_g = image_data[:, :, 1].flatten()
    image_data_b = image_data[:, :, 2].flatten()

    assert not np.isnan(image_data_r).any(), "Ada nilai NaN pada data
gambar merah"
    assert not np.isnan(image_data_g).any(), "Ada nilai NaN pada data
gambar hijau"
    assert not np.isnan(image_data_b).any(), "Ada nilai NaN pada data
gambar biru"

    return image_data_r, image_data_g, image_data_b

def save_model_pkl(filepath, hidden_weights, hidden_biases,
output_weights, output_biases):
    with open(filepath, 'wb') as file:
        pickle.dump((hidden_weights, hidden_biases, output_weights,
output_biases), file)
    print(f'Model saved to {filepath}')

def load_model_pkl(filepath):
    with open(filepath, 'rb') as file:
        hidden_weights, hidden_biases, output_weights, output_biases =
pickle.load(file)
    print(f'Model loaded from {filepath}')
    return hidden_weights, hidden_biases, output_weights, output_biases

# Fungsi untuk memeriksa NaN dalam array
def check_nan_in_weights_biases(hidden_weights, hidden_biases,
output_weights, output_biases):
    nan_in_hidden_weights = np.isnan(hidden_weights).any()
    nan_in_hidden_biases = np.isnan(hidden_biases).any()
    nan_in_output_weights = np.isnan(output_weights).any()
    nan_in_output_biases = np.isnan(output_biases).any()

    if nan_in_hidden_weights:
        print("NaN ditemukan dalam hidden_weights")
    if nan_in_hidden_biases:
        print("NaN ditemukan dalam hidden_biases")
    if nan_in_output_weights:
        print("NaN ditemukan dalam output_weights")
    if nan_in_output_biases:
        print("NaN ditemukan dalam output_biases")
```

```
    if    not    (nan_in_hidden_weights    or    nan_in_hidden_biases    or
nan_in_output_weights or nan_in_output_biases):
        print("Tidak ada NaN dalam bobot dan bias")

def check_extreme_values(weights, biases):
    # Check for NaN
    if np.isnan(weights).any() or np.isnan(biases).any():
        print("Ada nilai NaN dalam bobot atau bias.")
    # Check for extreme values
    if np.isinf(weights).any() or np.isinf(biases).any():
        print("Ada nilai inf dalam bobot atau bias.")
    if (weights > 1e100).any() or (biases > 1e100).any():
        print("Ada nilai yang sangat besar dalam bobot atau bias.")
    if (weights < -1e100).any() or (biases < -1e100).any():
        print("Ada nilai yang sangat kecil dalam bobot atau bias.")
```