Journal of Electrical Engineering and Computer (JEECOM)
Vol. xx, No. xx (20xx), DOI: 10.33650/jeecom.v4i2
p-ISSN: 2715-0410; e-ISSN: 2715-6427

Peningkatan Akurasi Deteksi Kendaraan Menggunakan Kombinasi Haar Cascade Classifier dan Convolutional Neural Networks (CNN)

Indra Irawanto[1], Andi Sunyoto[2], Kusnawi[3]
[1,2,3] Magister Teknik Informatika, Universitas Amikom Yogyakarta, Yogyakarta, Indonesia

Article Info

Article history:

Diterima 15 Februari 2024
Revisi 21 Februari 2024
Diterbitkan 5 April, 2024

Keywords:

Sistem Deteksi Mobil
Deep Learning
Convolutional Neural Network
Haar Cascade Classifier
Lampu Lalu Lintas Pintar

ABSTRAK

Teknologi pengolahan citra digital dan computer vision telah memainkan peran penting dalam meningkatkan sistem pengaturan lalu lintas. Meskipun kamera CCTV umum digunakan, kebanyakan sistem masih bersifat pasif dan terbatas dalam pengawasan arus lalu lintas. Dalam menanggapi kebutuhan akan sistem yang lebih proaktif dan adaptif, dikembangkan berbagai sistem Manajemen Lalu Lintas Pintar yang mengintegrasikan teknologi deteksi objek kendaraan canggih, seperti kombinasi Haar Cascade Classifier dengan Convolutional Neural Network (CNN). Haar Cascade Classifier efektif dalam mendeteksi objek real-time, namun dapat mengalami kesulitan dalam kondisi gambar kompleks. Integrasi dengan CNN diharapkan meningkatkan akurasi deteksi kendaraan dalam berbagai kondisi pencahayaan dan latar belakang. Penelitian ini bertujuan untuk mengeksplorasi arsitektur CNN yang optimal untuk diintegrasikan dengan Haar Cascade guna mencapai efisiensi dan akurasi deteksi kendaraan yang lebih tinggi dalam pengaturan lalu lintas. Dari hasil eksperimen, kombinasi Haar Cascade dan CNN efektif dalam mendeteksi dan mengestimasi jumlah kendaraan. Performa model tergantung pada kompleksitas gambar, di mana semakin kompleks gambar, semakin rendah akurasi dan sensitivitasnya. Penggunaan arsitektur MobileNet dan Xception menunjukkan kemampuan yang baik dalam mendeteksi kendaraan, dengan Xception memberikan sedikit peningkatan dalam akurasi **(80.13%)** dibandingkan dengan MobileNet **(79.19%)**, namun dengan waktu komputasi yang sedikit lebih lama (1.02 detik dibandingkan dengan 0.82 detik). Pilihan antara kedua model tergantung pada kebutuhan spesifik aplikasi, seperti kebutuhan untuk akurasi yang lebih tinggi atau kecepatan pemrosesan yang lebih cepat. Dengan demikian, penelitian ini berpotensi untuk memberikan kontribusi signifikan bagi pengembangan sistem lalu lintas yang lebih cerdas dan responsif di masa depan.

Corresponding Author:
Andi Sunyoto,
Universitas Amikom, Jl. Ring Road Utara, Ngringin, Condongcatur, Kec. Depok, Kabupaten Sleman, Daerah Istimewa Yogyakarta 55281, Indonesia
Email: andi@amikom.ac.id

1. PENDAHULUAN

Teknologi pengolahan citra digital dan *computer vision* telah mengalami kemajuan pesat dan banyak digunakan dalam berbagai konteks, termasuk dalam pengaturan lalu lintas. Penggunaan kamera CCTV (*Closed-Circuit Television*) telah menjadi umum di kota-kota besar, membantu petugas dalam mencegah pelanggaran dan memantau kondisi lalu lintas [1] . Namun, [2] menunjukkan bahwa sistem kamera CCTV tradisional lebih banyak berperan sebagai alat pemantau pasif, dengan kemampuan terbatas pada pengawasan arus lalu lintas saja. Ini memunculkan kebutuhan akan teknologi yang tidak hanya mendeteksi kendaraan secara otomatis tetapi juga dapat mengambil tindakan berdasarkan informasi yang dikumpulkan, seperti menghitung kepadatan lalu lintas atau mendeteksi pelanggaran

Untuk menjawab kebutuhan tersebut, telah dikembangkan berbagai sistem Manajemen Lalu Lintas Pintar. Sistem-sistem ini mengoptimalkan fungsi CCTV yang ada dengan teknologi deteksi objek kendaraan canggih untuk memantau dan mengatur lalu lintas secara dinamis, menyesuaikan durasi lampu lalu lintas berdasarkan kondisi lalu lintas aktual di persimpangan jalan [3]. Pendekatan pertama yang krusial adalah mengembangkan sistem deteksi kendaraan yang efektif, memanfaatkan pola dan fitur dalam citra kendaraan dengan algoritma *Deep Learning* seperti *Convolutional Neural Network* (CNN) [4]. CNN memanfaatkan prinsip kecerdasan buatan, dilatih dengan dataset besar untuk belajar dari data yang tersedia, dan bahkan memungkinkan penggunaan model pra-pelatihan melalui transfer learning untuk meningkatkan efisiensi [5].

Menghadapi tantangan dalam deteksi kendaraan, kami mengadopsi pendekatan yang menggabungkan *Haar Cascade Classifier* dengan *Convolutional Neural Network* (CNN) . *Haar Cascade Classifier*, yang menggunakan fitur persegi untuk memberikan indikasi spesifik pada gambar, dikenal karena kemampuannya dalam mendeteksi objek secara cepat dan real-time setelah melalui proses pelatihan [6]. Dengan semakin banyak sampel yang dipelajari, metode ini dapat meningkatkan tingkat akurasi pendeteksian secara signifikan [7]. Meskipun *Haar Cascade Classifier* efektif dalam mendeteksi objek dengan cepat, metode ini dapat mengalami kesulitan dalam kondisi gambar yang kompleks atau pencahayaan yang buruk. Oleh karena itu, kami mengintegrasikan *Haar Cascade* dengan teknologi CNN, yang memanfaatkan kecerdasan buatan untuk belajar dari sejumlah besar data dan meningkatkan kemampuan deteksi. Kombinasi kedua teknologi ini diharapkan tidak hanya meningkatkan akurasi deteksi kendaraan dalam berbagai kondisi pencahayaan dan *background* tetapi juga memungkinkan pengolahan gambar dan video lalu lintas secara real-time [8]. Dalam penelitian ini, kami akan mengeksplorasi arsitektur CNN yang optimal untuk mengintegrasikan dengan *Haar Cascade*, dengan tujuan mencapai efisiensi dan akurasi deteksi kendaraan yang lebih tinggi.

Penelitian ini disusun dalam tiga bagian utama. Bagian II akan menguraikan metode yang mencakup formula atau rumusan yang diterapkan dalam analisis sentimen. Bagian III akan menampilkan hasil dan diskusi dari penerapan metode tersebut. Bagian IV akan menyimpulkan temuan dari penelitian yang telah dilakukan.

2. METODE

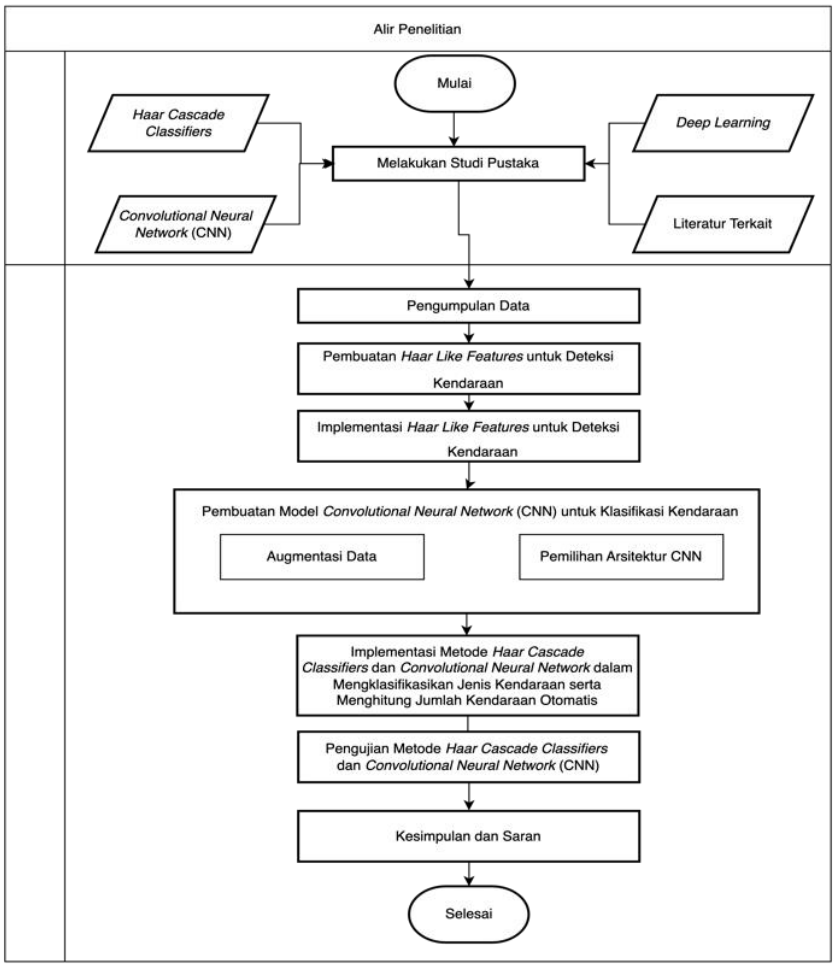
Penelitian ini memanfaatkan sinergi antara *Haar Cascade Classifier* dan *Convolutional Neural Network* (CNN) untuk meningkatkan deteksi dan penghitungan kendaraan dalam cuplikan video CCTV. Setiap metode membawa kelebihan tersendiri yang berkontribusi pada peningkatan presisi dan efisiensi dalam identifikasi kendaraan.

Haar cascade classifier adalah metode yang mengidentifikasi fitur-fitur persegi panjang secara spesifik dalam sebuah gambar atau citra. Metode ini berasal dari gagasan Paul Viola dan Michael Jhon, sehingga dikenal sebagai metode Viola & Jhon [9]. *Haar Cascade Classifier* menonjol dalam kemampuannya untuk mendeteksi objek dengan kecepatan dan efisiensi yang tinggi. Metode ini berhasil dengan cara mengenali fitur-fitur kunci dalam sebuah gambar yang dikenal sebagai *Haar-like features*, yang mempercepat proses deteksi. Kecepatan ini sangat penting untuk pemrosesan waktu nyata dalam video CCTV, dimana respons cepat esensial untuk pengelolaan lalu lintas yang berubah-ubah.

Di sisi lain, Convolutional Neural Network (CNN) sering dipilih dalam pemrosesan citra karena memiliki tingkat akurasi yang tinggi dan kemampuan yang lebih baik dalam mengenali gambar visual [10]. Penerapan CNN meningkatkan kemampuan dalam mengenali objek dengan akurasi yang lebih tinggi. CNN mampu menginterpretasikan struktur dan fitur kompleks dalam gambar, memungkinkan identifikasi kendaraan yang tepat dengan tingkat keakuratan yang lebih tinggi. Dengan memanfaatkan kekuatan dari deep learning, CNN mengatasi tantangan yang sering dihadapi dalam deteksi objek di luar ruangan, seperti variabilitas pencahayaan dan perspektif. Namun CNN, seperti metode Deep Learning lainnya, memiliki kelemahan yaitu proses pelatihan model yang cukup lama .

Integrasi dari *Haar Cascade Classifier* dan CNN menggabungkan deteksi cepat dan akurasi pengenalan tinggi, secara signifikan meningkatkan kinerja sistem dalam mendeteksi dan menghitung kendaraan dalam video CCTV. Selain itu, penelitian ini mengeksplorasi perbandingan arsitektur deep learning Xception dan MobileNet untuk mengetahui manakah yang lebih efektif dalam konteks deteksi kendaraan. Kami menilai kedua arsitektur tidak hanya berdasarkan presisi tetapi juga berdasarkan efisiensi waktu komputasi, yang penting untuk implementasi dalam skenario waktu nyata.

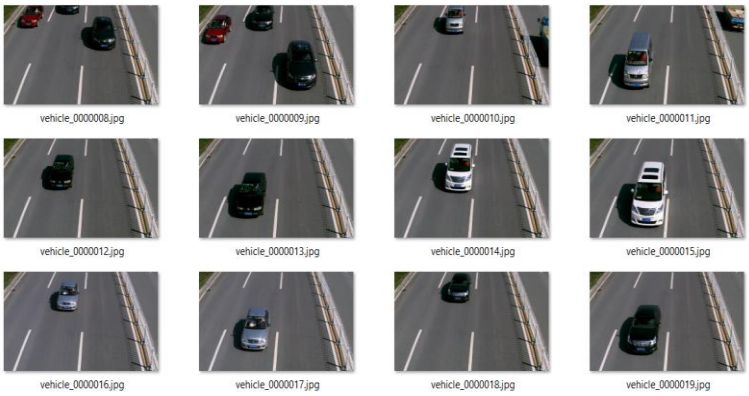
Penelitian ini juga mencakup serangkaian langkah metodis sebagaimana digambarkan dalam model penelitian kami, dimulai dari pengumpulan data hingga penarikan kesimpulan, yang akan diuraikan lebih lanjut dan divisualisasikan dalam Gambar 1.



Gambar 1. Langkah-langkah penelitian

2.1. Pengumpulan Data

Proses pengumpulan data dalam penelitian ini menggunakan dataset BIT-Vehicle, yang terdiri dari sejumlah gambar kendaraan yang diambil menggunakan dua kamera pada berbagai waktu dan lokasi. Dimensi gambar bervariasi antara 1600x1200 dan 1920x1080 piksel. Dataset ini mencakup variasi dalam kondisi pencahayaan, skala, warna permukaan kendaraan, serta sudut pandang. Beberapa gambar mungkin tidak menampilkan seluruh bagian kendaraan karena keterlambatan dalam pengambilan gambar dan perbedaan ukuran kendaraan. Setiap gambar dapat menampilkan satu atau dua kendaraan, dengan lokasi masing-masing kendaraan telah diannotasikan sebelumnya. Kendaraan dalam dataset ini terbagi menjadi enam kategori: Bus, Mikrobus, Minivan, Sedan, SUV, dan Truk, dengan jumlah kendaraan per kategori berturut-turut adalah 558, 883, 476, 5.922, 1.392, dan 822. Dataset ini akan menjadi sumber informasi yang berharga dalam pengembangan dan penelitian terkait deteksi serta identifikasi kendaraan dalam berbagai kondisi. Gambar 2 merupakan contoh dari dataset yang kami gunakan.

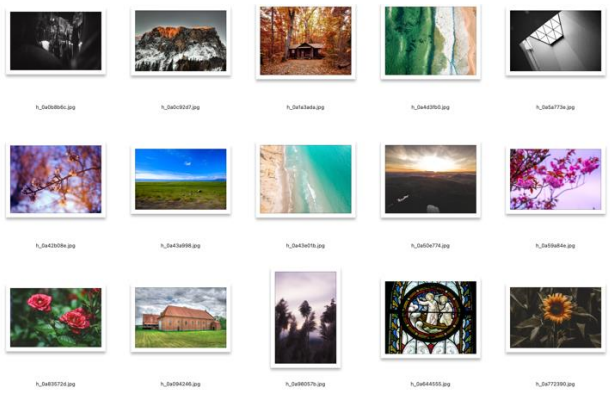


Gambar 2. Dataset Kendaraan

Kami juga memperoleh data video dari kamera pengawas lalu lintas di platform YouTube dengan mencari kata kunci seperti "CCTV traffic", "roads", "cars", dan "highway". Sebanyak 12 video dengan format file *.mp4 berhasil kami temukan dari hasil pencarian tersebut, yang kemudian kami ekstraksi dan gunakan sebagai dataset tambahan. Gambar 3 merupakan contoh dari video CCTV yang kami gunakan dalam penelitian ini.

Gambar 3. Data Video CCTV

Untuk melengkapi dataset kendaraan kami, kami telah menambahkan citra non-mobil dari dataset pemandangan yang dikumpulkan oleh [11]. Dataset ini berisi sekitar 26.000 gambar pemandangan dengan resolusi rendah. Untuk keperluan penelitian ini, kami telah memilih secara selektif 10.000 citra pemandangan dari koleksi tersebut. Gambar 4 menampilkan contoh citra pemandangan yang telah kami seleksi untuk digunakan dalam penelitian kami.

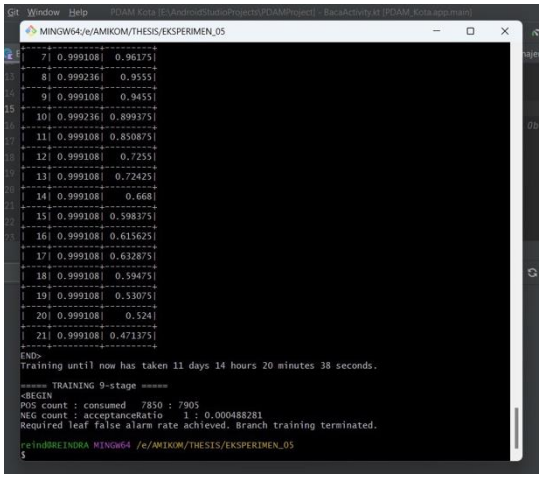


Gambar 4. Dataset bukan mobil

2.2. Membuat *Haar Like Features* Deteksi Mobil

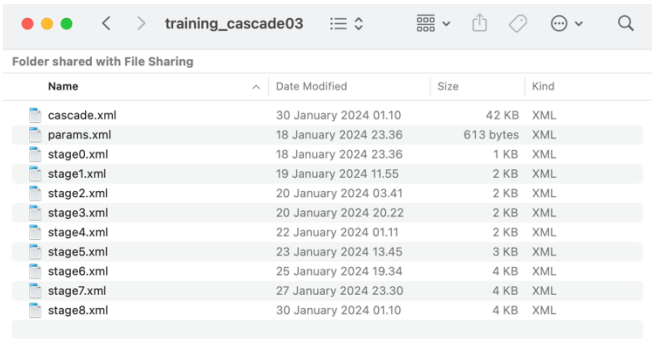
Proses pembuatan *Features Haar Like* untuk deteksi mobil menggunakan *OpenCV*, langkah-langkahnya diawali dengan pembuatan file annotasi.txt yang berisi informasi lokasi dan ukuran objek mobil pada gambar. Selanjutnya, dilakukan proses pembuatan detector negative untuk mengumpulkan gambar-gambar non-mobil yang tidak diinginkan. Setelah itu, dilakukan proses pembuatan file vektor dengan menggunakan alat bawaan OpenCV, yang mengonversi gambar-gambar pelatihan menjadi format yang dapat digunakan oleh algoritma pelatihan Cascade Classifier.

Langkah berikutnya adalah melatih *Haar Like Features* dengan menggunakan alat pelatihan *Cascade Classifier* yang disediakan oleh OpenCV [6]. Proses pelatihan ini mencakup penentuan sampel positif dan negatif, di mana sampel positif mencakup gambar-gambar mobil yang telah dianotasi, sedangkan sampel negatif mencakup gambar-gambar non-mobil yang telah didefinisikan. Setelah menentukan sampel, model dilatih dengan menggunakan algoritma pembelajaran mesin untuk mengidentifikasi fitur-fitur Haar [12] yang mewakili objek mobil. Proses pembuatan Features Haar Like memerlukan waktu sekitar 11 hari, tergantung pada ukuran dataset dan spesifikasi perangkat keras yang digunakan. Gambar 5 merupakan gambaran dari proses pembuatan *Features Haar Like*.



Gambar 5. Proses Pembuatan Haar Like Features Deteksi Mobil

Setelah proses pelatihan selesai, klasifier yang dilatih dievaluasi kinerjanya dengan menggunakan dataset uji yang telah disiapkan sebelumnya. Alasan pemilihan OpenCV untuk langkah-langkah ini adalah karena OpenCV menyediakan algoritma dan fungsi yang dioptimalkan untuk pemrosesan citra dan deteksi objek. Alat-alat yang disediakan oleh OpenCV memudahkan proses pengembangan dan implementasi sistem deteksi mobil, serta memastikan konsistensi dan keandalan dalam hasil deteksi. Dengan demikian, penggunaan OpenCV sebagai kerangka kerja utama dalam pengembangan *Haar Like Features* untuk deteksi mobil menjadi pilihan yang rasional dan efisien. Gambar 6 merupakan hasil dari proses pembuatan *Haar Like Features*.

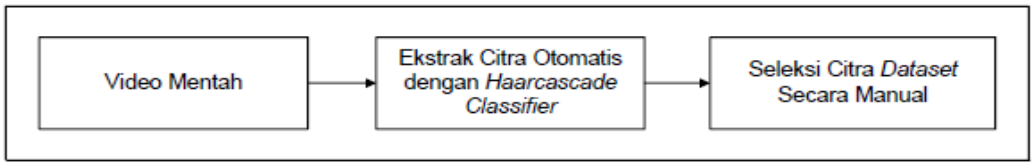


Gambar 6. Hasil dari proses pembuatan Haar Like Features

2.3. Implementasi Filter *Haar Cascade Classifier* untuk Deteksi Mobil Pada CCTV

Pada langkah ini, *Haar Cascade classifiers* yang berhasil dikembangkan akan digunakan untuk mengidentifikasi objek mobil dalam rekaman video dari CCTV. Hasil deteksi ini akan membentuk dataset gambar mobil dan bukan mobil yang akan digunakan dalam pembuatan model Convolutional Neural Network (CNN). Proses ekstraksi video dilakukan secara otomatis dengan menggunakan Haar Cascade classifiers pada data video yang diambil dari *platform* YouTube. Filter ini diimplementasikan dalam program Python untuk otomatis mendeteksi mobil dan mengekstraksi gambar. Setelah proses ekstraksi

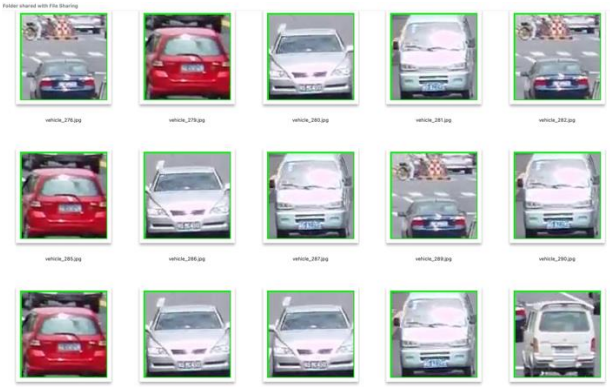
video selesai, gambar yang telah diekstraksi akan dipilih secara manual untuk memastikan kualitasnya sesuai dengan kebutuhan. Alur dari tahap ini dapat dilihat pada Gambar 7.



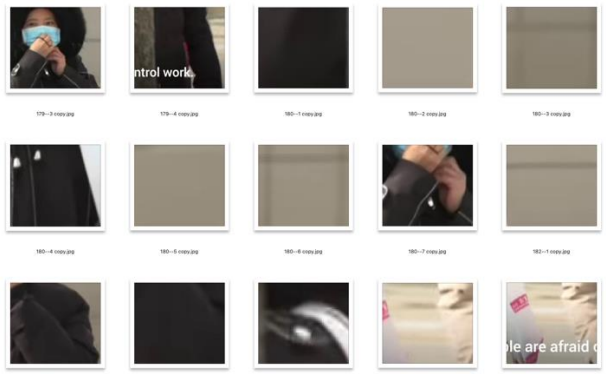
Gambar 7. Alur tahapan ekstraksi video CCTV

2.4. Pembuatan Model *Convolutional Neural Network* (CNN) Untuk Deteksi Mobil

Dalam penelitian ini, kami melakukan tahap pembuatan model Convolutional Neural Network (CNN) untuk deteksi mobil setelah proses ekstraksi data dari rekaman video CCTV menggunakan filter Haar Cascade classifiers. Hasil dari proses ekstraksi tersebut adalah dataset gambar mobil dan bukan mobil yang akan digunakan dalam pembuatan model CNN. Kami telah mengumpulkan secara manual total keseluruhan 2178 gambar mobil dan 2532 gambar non-mobil, yang kemudian dibagi menjadi tiga bagian data: training, testing, dan validation. Sebagai contoh, pada Gambar 8 dan 9 di bawah ini, dapat dilihat representasi visual dari kedua kelas data latih yang digunakan. Gambar 8 menampilkan beberapa contoh citra mobil yang termasuk dalam kategori pertama, sementara Gambar 9 menunjukkan citra-citra yang merupakan bagian dari kategori kedua, yaitu bukan mobil.



Gambar 8. Dataset mobil hasil seleksi manual



Gambar 9. Dataset bukan mobil hasil seleksi manual

Dalam pengembangan model CNN, kami menggunakan TensorFlow dan mengimpor library yang diperlukan serta melakukan konfigurasi umum, termasuk penentuan ukuran gambar, direktori data latih, validasi, dan pengujian, serta jumlah epoch dan ukuran batch. Pada tahap inisialisasi model, saya memilih dua arsitektur, yaitu MobileNet dan Xception, sebagai perbandingan. Kami menambahkan lapisan-lapisan tambahan di atasnya, seperti Global Average Pooling, lapisan Dense, dan lapisan Dropout untuk

mengurangi overfitting. Selanjutnya, saya membekukan layer-layer pada base model agar bobot yang telah dipelajari tidak diubah selama proses pelatihan model.

Setelah proses inisialisasi, model CNN disusun dengan menentukan optimizer, fungsi loss, dan metrik yang akan dimonitor. Kemudian, proses augmentasi dilakukan pada data latih dan validasi menggunakan ImageDataGenerator [13]. Setelah itu, generator dibuat untuk kedua dataset tersebut. Model CNN kemudian dilatih menggunakan data latih [14]. Setelah proses pelatihan selesai, model dinilai menggunakan data pengujian.

Terakhir, model CNN yang telah dilatih disimpan dalam format .h5 [15] untuk digunakan di masa mendatang tanpa perlu melatih ulang. Proses ini mewakili siklus umum dalam pembuatan model CNN, yang dimulai dari inisialisasi hingga evaluasi dan penyimpanan model yang telah dilatih. Gambar 10 merupakan hasil dari proses pembuatan model CNN.

```
Found 3608 images belonging to 2 classes.
Found 551 images belonging to 2 classes.
Epoch 1/50
225/225 [==============================] - 112s 488ms/step - loss: 0.1249 - accuracy: 0.9674 - val_loss: 0.0432 - val_accuracy: 0.9835
Epoch 2/50
225/225 [==============================] - 110s 488ms/step - loss: 0.0541 - accuracy: 0.9869 - val_loss: 0.0706 - val_accuracy: 0.9816
Epoch 3/50
225/225 [==============================] - 104s 463ms/step - loss: 0.0369 - accuracy: 0.9919 - val_loss: 0.0667 - val_accuracy: 0.9835
Epoch 4/50
225/225 [==============================] - 100s 445ms/step - loss: 0.0319 - accuracy: 0.9919 - val_loss: 0.1275 - val_accuracy: 0.9835
Epoch 5/50
225/225 [==============================] - 99s 442ms/step - loss: 0.0382 - accuracy: 0.9942 - val_loss: 0.0778 - val_accuracy: 0.9871
Epoch 6/50
225/225 [==============================] - 4889s 22s/step - loss: 0.0328 - accuracy: 0.9950 - val_loss: 0.2941 - val_accuracy: 0.9596
Epoch 7/50
225/225 [==============================] - 113s 504ms/step - loss: 0.0347 - accuracy: 0.9922 - val_loss: 0.1193 - val_accuracy: 0.9761
Epoch 8/50
225/225 [==============================] - 109s 484ms/step - loss: 0.0279 - accuracy: 0.9961 - val_loss: 0.1224 - val_accuracy: 0.9835
Epoch 9/50
225/225 [==============================] - 110s 488ms/step - loss: 0.0284 - accuracy: 0.9942 - val_loss: 0.0539 - val_accuracy: 0.9890
Epoch 10/50
225/225 [==============================] - 6421s 29s/step - loss: 0.0356 - accuracy: 0.9950 - val_loss: 0.0826 - val_accuracy: 0.9871
Epoch 11/50
225/225 [==============================] - 110s 488ms/step - loss: 0.0217 - accuracy: 0.9950 - val_loss: 0.0892 - val_accuracy: 0.9890
Epoch 12/50
...
225/225 [==============================] - 103s 459ms/step - loss: 0.0145 - accuracy: 0.9975 - val_loss: 0.1992 - val_accuracy: 0.9816
Found 551 images belonging to 2 classes.
34/34 [==============================] - 13s 384ms/step - loss: 0.1017 - accuracy: 0.9890
Test Loss: 0.10172994434833527, Test Accuracy: 0.9889705777168274
```

Gambar 10. Dataset bukan mobil hasil seleksi manual

2.5. Implementasi Metode Haar Cascade Classifiers dan Convolutional Neural Network dalam Menghitung Jumlah Mobil Otomatis

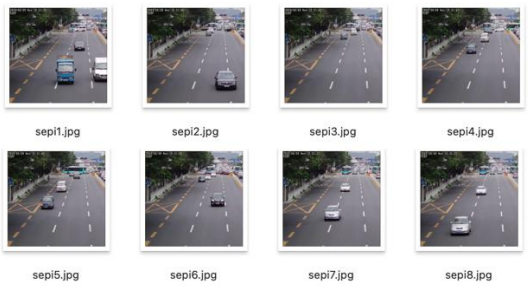
Pada tahapan implementasi, kami menggabungkan pendekatan *Haar Cascade Classifiers* dengan model CNN yang telah dikonfigurasi sebelumnya untuk mengidentifikasi dan menghitung jumlah kendaraan. Kami menggunakan Python versi 3.10.9 sebagai bahasa pemrograman, dengan eksekusi pada lingkungan Visual Studio Code. Library penting seperti OpenCV python, NumPy, TensorFlow, dan Keras diintegrasikan untuk mendukung pengembangan algoritma ini.

Awalnya, *Haar Cascade Classifiers* digunakan untuk mendeteksi kemungkinan keberadaan kendaraan dalam gambar. Setiap deteksi yang berhasil kemudian di pangkas dan disimpan sebagai file gambar sementara. Hasil ini kemudian diproses oleh model CNN yang telah di latih untuk validasi lebih lanjut, dengan tujuan untuk mengkonfirmasi identifikasi kendaraan. Setelah konfirmasi oleh model CNN, setiap deteksi yang valid di highlight dengan kotak persegi berwarna biru pada gambar asli, dan waktu komputasi serta jumlah total kendaraan dihitung sebagai indikator kepadatan lalu lintas.

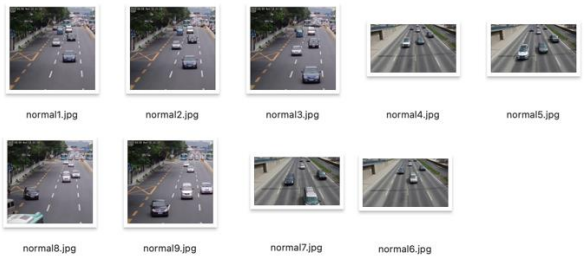
2.6. Uji Coba

Dalam fase pengujian, evaluasi performa sistem telah dilaksanakan menggunakan serangkaian gambar uji yang berasal dari rekaman CCTV untuk tiga skenario lalu lintas: kondisi sepi, normal, dan ramai. Gambar-gambar ini telah diklasifikasikan sesuai dengan keadaan lalu lintas yang terpantau pada saat pengambilan gambar. Untuk tujuan penelitian ini, Pengujian ini melibatkan 8 gambar untuk kondisi sepi, 9 gambar untuk kondisi normal, dan 6 gambar untuk kondisi ramai, menjadikan total 23 gambar yang digunakan sebagai sampel uji. Kinerja metode dinilai berdasarkan efisiensi waktu komputasi dan akurasi dalam mengidentifikasi dan menghitung jumlah mobil.

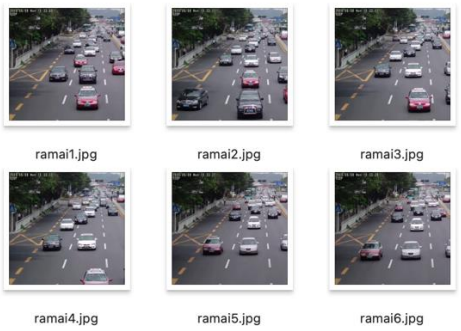
Evaluasi dari sistem ini diukur dengan mempertimbangkan dua aspek kinerja utama: waktu komputasi yang diperlukan untuk memproses gambar dan tingkat keakuratan dalam mendeteksi dan menghitung jumlah mobil. Ini memberikan indikasi tentang efisiensi sistem dalam mengelola sumber daya komputasi dan keandalannya dalam menginterpretasikan berbagai kondisi lalu lintas secara akurat. Data yang dihasilkan dari pengujian ini akan memberikan penilaian yang objektif terhadap performa keseluruhan metode yang diintegrasikan dalam penelitian ini. Gambar 11 sampai Gambar 13 merupakan contoh data uji yang digunakan pada masing masing kategori.



Gambar 11. Data uji kondisi sepi



Gambar 12. Data uji kondisi sepi



Gambar 13. Data uji kondisi ramai

3. HASIL DAN PEMBAHASAN

Pada tahap hasil, kita mengamati dan menilai performa sistem berdasarkan data uji. Setiap gambar diuji menggunakan algoritme yang telah dijelaskan, dan hasilnya dicatat dengan detail. Kinerja sistem dievaluasi dengan memperhatikan jumlah deteksi yang akurat dan kecepatan proses komputasi. Diskusi selanjutnya menggali lebih dalam tentang pengaruh kondisi lalu lintas terhadap tingkat keberhasilan deteksi, serta mencari tahu penyebab dari setiap kesalahan klasifikasi yang mungkin terjadi.

Pada tahap ini, hasil eksperimen akan dianalisis untuk menentukan tingkat keberhasilan metode yang digunakan. Dua poin diskusi akan disajikan, yaitu analisis hasil uji coba metode Haar Cascade Classifiers dan Convolutional Neural Network (CNN) pada data uji kondisi sepi, normal, dan ramai.

3.1. Uji Coba Metode *Haar Cascade Classifiers* dan *Convolutional Neural Network*

Hasil dari metode ini berupa gambar yang menunjukkan objek yang berhasil terdeteksi beserta tingkat akurasi dan perkiraan jumlah kendaraannya. Contoh hasil deteksi menggunakan metode Haar Cascade Classifiers dan Convolutional Neural Network dapat dilihat pada Gambar 14 sampai Gambar 16.

Gambar 14. Hasil uji coba dalam kondisi ramai

Gambar 15. Hasil uji coba dalam kondisi ramai

Gambar 16. Hasil uji coba dalam kondisi sepi

Dari evaluasi yang telah dilakukan, terlihat bahwa metode deteksi yang diimplementasikan berhasil mengidentifikasi sebagian besar kendaraan pada gambar. Namun, terdapat beberapa kendaraan yang tidak terdeteksi, khususnya kendaraan yang berlokasi lebih jauh dari kamera serta kendaraan yang berada dalam satu garis atau tumpang tindih, yang mana ini menunjukkan batasan dari metode yang digunakan. Kendaraan yang lebih jauh cenderung lebih kecil dalam frame gambar, yang bisa menyulitkan algoritme Haar Cascade untuk mendeteksi ciri-ciri kendaraan. Demikian pula, kendaraan yang bertumpukan bisa menyebabkan beberapa kendaraan dianggap sebagai satu objek, yang mengurangi kemampuan sistem untuk menghitung setiap kendaraan secara terpisah. Table dibawah ini merupakan hasil dari pengujian dengan arsitektur MobileNet dan Xception.

Tabel 1. Hasil Pengujian MobileNet

TABLE I. HASIL PENGUJIAN DENGAN MOBILENET

Nama File	Total Objek	Nilai Prediksi			Akurasi	Waktu Komputasi
		TP	FP	FN		
Normal1.jpg	6	6	0	0	100%	0.83 detik
Normal2.jpg	5	5	0	0	100%	0.70 detik
Normal3.jpg	5	5	0	0	100%	0.72 detik
Normal4.jpg	7	5	0	2	71.43%	1.34 detik
Normal5.jpg	7	5	0	2	71.43%	0.70 detik
Normal6.jpg	6	4	0	2	66.67%	0.56 detik
Normal7.jpg	5	3	0	2	60.67%	0.61 detik
Normal8.jpg	5	4	0	1	80.00%	0.81 detik
Normal9.jpg	5	5	0	0	100%	0.84 detik
Ramai5.jpg	14	9	0	5	64.29%	1.42 detik
Ramai6.jpg	15	10	0	5	66.67%	1.58 detik
Ramai7.jpg	15	9	0	6	60.00%	1.33 detik
Ramai8.jpg	14	9	0	5	64.29%	1.36 detik
Ramai9.jpg	12	9	0	3	75.00%	1.39 detik
Ramai10.jpg	12	8	0	4	66.67%	1.29 detik
Sepi3.jpg	3	3	0	0	100%	0.52 detik
Sepi4.jpg	2	1	0	1	50.00%	0.34 detik
Sepi5.jpg	2	1	0	1	50.00%	0.44 detik
Sepi6.jpg	2	2	0	0	100%	0.50 detik
Sepi7.jpg	3	3	0	0	100%	0.45 detik
Sepi8.jpg	4	3	0	1	75.00%	0.47 detik
Sepi9.jpg	2	2	0	0	100%	0.32 detik
Sepi10.jpg	2	2	0	0	100%	0.39 detik
Rata - rata					79.19%	0.82 detik

Tabel 2. Hasil Pengujian Xception

TABLE II. HASIL PENGUJIAN DENGAN XCEPTION

Nama File	Total Objek	Nilai Prediksi			Akurasi	Waktu Komputasi
		TP	FP	FN		
Normal1.jpg	6	6	0	0	100%	1.17 detik
Normal2.jpg	5	5	0	0	100%	1.26 detik
Normal3.jpg	5	5	0	0	100%	1.08 detik
Normal4.jpg	7	5	0	2	71.43%	1.57 detik
Normal5.jpg	7	5	0	2	71.43%	0.85 detik
Normal6.jpg	6	4	0	2	66.67%	0.76 detik
Normal7.jpg	5	3	0	2	60.67%	0.62 detik
Normal8.jpg	5	4	0	1	80.00%	1.00 detik
Normal9.jpg	5	5	1	0	100%	0.84 detik
Ramai5.jpg	14	9	1	5	64.29%	1.58 detik
Ramai6.jpg	15	11	0	4	73.33%	2.13 detik
Ramai7.jpg	15	10	0	5	66.67%	1.53 detik
Ramai8.jpg	14	9	0	5	64.29%	1.57 detik
Ramai9.jpg	12	9	0	3	75.00%	1.58 detik
Ramai10.jpg	12	9	0	3	75.00%	1.72 detik
Sepi3.jpg	3	3	0	0	100%	0.71 detik
Sepi4.jpg	2	1	0	1	50.00%	0.39 detik
Sepi5.jpg	2	1	0	1	50.00%	0.45 detik
Sepi6.jpg	2	2	0	0	100%	0.46 detik
Sepi7.jpg	3	3	0	0	100%	0.59 detik
Sepi8.jpg	4	3	1	1	100%	0.56 detik
Sepi9.jpg	2	2	0	0	100%	0.57 detik
Sepi10.jpg	2	2	0	0	100%	0.52 detik
Rata - rata					80.13%	1.02 detik

4. KESIMPULAN

Berdasarkan analisis dari eksperimen yang telah dilakukan, kesimpulan yang dapat ditarik menunjukkan bahwa kombinasi metode Haar Cascade dan Convolutional Neural Network (CNN) cukup efektif dalam mendeteksi dan mengestimasi jumlah kendaraan. Dari hasil pengujian, terlihat bahwa tingkat akurasi dan sensitivitas model berfluktuasi berdasarkan kategori data uji yang berbeda. Hasil tersebut menegaskan bahwa kompleksitas gambar, yang diukur dari jumlah objek dan tumpang tindihnya, berbanding terbalik dengan performa model; dengan kata lain, semakin kompleks gambar, semakin rendah akurasi dan sensitivitas yang dicapai oleh model. Dari hasil pengujian, kedua arsitektur juga menunjukkan kemampuan yang baik dalam mendeteksi mobil dalam berbagai kondisi lalu lintas. Rata-rata akurasi dari arsitektur MobileNet adalah 79.19% dengan waktu komputasi rata-rata 0.82 detik. Sementara itu, penggunaan arsitektur Xception meningkatkan akurasi rata-rata menjadi 80.13%, namun dengan waktu komputasi yang sedikit lebih lama, yaitu rata-rata 1.02 detik. Ini menunjukkan bahwa sementara Xception mungkin memberikan sedikit peningkatan dalam akurasi, hal ini datang dengan biaya komputasi yang lebih tinggi. Pilihan antara kedua model tersebut mungkin akan bergantung pada kebutuhan spesifik aplikasi, seperti kebutuhan untuk akurasi yang lebih tinggi atau kebutuhan untuk kecepatan pemrosesan yang lebih cepat.

Untuk penelitian di masa depan, sangat disarankan untuk mengembangkan dan melatih model dengan dataset yang lebih luas dan bervariasi, yang mencakup berbagai skenario lalu lintas dan kondisi cuaca yang berbeda. Hal ini diharapkan akan meningkatkan kemampuan adaptasi model terhadap beragam kondisi visual. Proses augmentasi data yang lebih intensif dapat pula meningkatkan robustness model terhadap variasi dalam gambar yang lebih luas.

Selanjutnya, akan bermanfaat untuk memperluas jumlah kategori dalam lalu lintas yang diuji, termasuk kategori pejalan kaki, sepeda, dan berbagai jenis kendaraan seperti mobil, truk, dan bus. Ini akan memberikan ujian yang lebih komprehensif terhadap daya tahan model dalam skenario yang lebih kompleks, yang pada gilirannya akan memperluas aplikasinya dalam sistem transportasi. Khususnya, menambahkan kategori kendaraan yang lebih spesifik seperti truk dan bus akan memberikan gambaran yang lebih mendetail tentang komposisi lalu lintas.

Penelitian selanjutnya juga harus berfokus pada perbaikan metode deteksi, terutama dalam mengatasi tantangan deteksi kendaraan yang tumpang tindih atau yang berada pada jarak yang lebih jauh dari kamera. Mengeksplorasi arsitektur CNN yang lebih canggih dan teknik transfer learning yang lebih baru bisa menjadi langkah yang strategis untuk meningkatkan akurasi dan efisiensi deteksi. Pemilihan fitur Haar yang lebih sensitif dan adaptif juga penting untuk diteliti guna mengatasi batasan yang ada saat ini.

Melalui peningkatan ini, diharapkan penelitian mendatang dapat lebih menyempurnakan sistem pendeteksian kendaraan berbasis deep learning, sehingga lebih efektif dalam mengatasi tantangan kemacetan lalu lintas dan secara signifikan meningkatkan efisiensi sistem transportasi.

UCAPAN TERIMA KASIH

Kami, sebagai penulis, ingin mengucapkan rasa terima kasih kepada pembimbing kami, yaitu Dr. Andi Sunyoto, M. Kom., dan Kusnawi, S.Kom., M.Eng., atas bimbingan, dukungan, dan motivasi yang telah diberikan selama proses penelitian ini.

REFERENSI

[1] R. G. Fajri, I. Santoso, Y. Alvin, and A. Soetrisno, "PERANCANGAN PROGRAM PENDETEKSI DAN PENGKLASIFIKASI JENIS KENDARAAN DENGAN METODE CONVOLUTIONAL NEURAL NETWORK (CNN) DEEP LEARNING," 2020. [Online]. Available: https://ejournal3.undip.ac.id/index.php/transient

[2] M. Aghassi Zulfikar, M. Somantri, and D. Sudjadi, "PENERAPAN METODE CONVOLUTIONAL NEURAL NETWORK (CNN) DAN LONG SHORT TERM MEMORY (LSTM) UNTUK PENGENALAN AKTIVITAS MANUSIA PADA CCTV DI AREA TAMBAK UDANG," 2021. [Online]. Available: https://ejournal3.undip.ac.id/index.php/transient

[3] M. Hasanah *et al.*, "Automatic Car Detection Using Haar Cascade Classifier and Convolutional Neural Network for Traffic Density Estimation," *Indonesian Journal of Artificial Intelligence and Data Mining (IJAIDM)*, vol. 4, no. 1, pp. 11–18, 2021, doi: 10.24014/ijaidm.v4i1.10785.

[4] A. Gholamhosseinian and J. Seitz, "Vehicle Classification in Intelligent Transport Systems: An Overview, Methods and Software Perspective," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 2. Institute of Electrical and Electronics Engineers Inc., pp. 173–194, 2021. doi: 10.1109/OJITS.2021.3096756.

[5] M. Athoillah and R. K. Putri, "IDENTIFIKASI JENIS KENDARAAN BERMOTOR DENGAN ALGORITMA CONVOLUTIONAL NEURAL NETWORKS," *VARIANCE: Journal of Statistics and Its Applications*, vol. 5, no. 2, pp. 109–116, Oct. 2023, doi: 10.30598/variancevol5iss2page109-116.

[6] P. Kenda and A. Witanti, "Sistem Presensi Berbasis Wajah Dengan Metode Haar Cascade," *Konvergensi Teknologi dan Sistem Informasi*, 2021.

[7] J. Sitompul, M. I. Bustami, and D. Kisbianty, "Implementasi Algoritma Haar Cascade Classifier Dalam Mendeteksi Robot Sepak Bola Beroda," *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 6, no. 4, p. 2032, Oct. 2022, doi: 10.30865/mib.v6i4.3929.

[8] M. Singh Bhatia, A. Aggarwal, and N. Kumar, "Smart Traffic Light System to Control Traffic Congestion PJAEE, 17 (9) (2020) Smart Traffic Light System to Control Traffic Congestion," *Palarch's Journal Of Archaeology Of Egypt/Egyptology*, 2020.

[9] M. F. Mustaqim, A. Nugroho, D. Alfa, and F. Suni, "Sistem Deteksi Kecepatan Kendaraan Menggunakan Metode Haar Cascade untuk Keamanan Berkendara," *Edu Elektrika Journal*, vol. 10, no. 2.

[10] S. Yuliany and A. Nur Rachman, "Implementasi Deep Learning pada Sistem Klasifikasi Hama Tanaman Padi Menggunakan Metode Convolutional Neural Network (CNN)," 2022.

[11] A. Oliva and A. Torralba, "Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope *," 2001.

[12] R. Darmawan, "Perancangan Sistem Absensi menggunakan Face Recognition dengan Haar Cascade Classifier," 2022.

[13] I. Patresia Tambun, "KLASIFIKASI ALFABET BAHASA ISYARAT INDONESIA (BISINDO) DENGAN MENGGUNAKAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK," 2023.

[14] Ulfah Nur Oktaviana, Ricky Hendrawan, Alfian Dwi Khoirul Annas, and Galih Wasis Wicaksono, "Klasifikasi Penyakit Padi berdasarkan Citra Daun Menggunakan Model Terlatih Resnet101," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 5, no. 6, pp. 1216–1222, Dec. 2021, doi: 10.29207/resti.v5i6.3607.

[15] "Penghindaran Rintangan Otomatis Pada AgenOtonomBerbasis End-to-End Deep Imitation Learning".