IMPLEMENTASI CONVOLUTIONAL NEURAL NETWORK (CNN) UNTUK DETEKSI MARKA JALAN

Farhan Maulana Ridho, Ahmad Fahrudi Setiawan, Joseph Dedy Irawan
Teknik Informatika, Institut Teknologi Nasional Malang
Jalan Raya Karanglo km 2 Malang, Indonesia
2118139@scholar.itn.ac.id

ABSTRAK

Deteksi marka jalan merupakan komponen kritis dalam pengembangan sistem kendaraan otonom serta sistem bantuan pengemudi modern. Meskipun demikian, akurasi dalam proses deteksi masih menghadapi tantangan, terutama dalam kondisi jalan berbelok, tanda jalan yang tidak jelas, atau kondisi pencahayaan yang kurang optimal. Penelitian ini bertujuan untuk merancang sistem deteksi marka jalan berbasis *Convolutional Neural Network* (CNN) yang diterapkan pada video dashcam mobil dan terintegrasi ke dalam aplikasi web menggunakan framework Flask. Dataset yang digunakan terdiri dari video dashcam yang diproses per frame menggunakan OpenCV, kemudian diklasifikasi dengan model CNN berarsitektur VGG16. Hasil deteksi ditampilkan dalam bentuk gambar serta video overlay yang dapat diunduh dan disimpan di *cloud Supabase*. Pengujian fungsional menunjukkan sistem berhasil memvalidasi input dan menampilkan hasil tanpa error. Pengujian dilakukan pada sepuluh video dengan durasi 60 detik dan rata-rata FPS sekitar 40–41, dengan waktu pemrosesan berkisar antara 115–142 detik. Performa deteksi model juga terbukti sangat dipengaruhi oleh kondisi jalan, di mana hasilnya cenderung kurang stabil pada skenario jalan berbelok akibat distorsi perspektif marka yang signifikan. Dari sisi kinerja, sistem mampu memproses video dengan kecepatan rata-rata 14.0 FPS, yang memadai untuk analisis offline namun belum memenuhi standar real-time 25-30 FPS. Sistem menunjukkan performa baik pada kondisi jalan lurus dan marka yang jelas, namun masih memerlukan peningkatan untuk kondisi jalan yang kompleks. Penelitian ini memberikan kontribusi sebagai fondasi awal dalam pengembangan sistem deteksi jalur otomatis yang lebih adaptif dan responsif terhadap berbagai skenario lalu lintas.

Kata kunci : *Deteksi Jalur Jalan, Convolutional Neural Network, Deep Learning, Flask, Kendaraan Otonom.*

1. PENDAHULUAN

Jalan merupakan prasarana transportasi darat yang meliputi segala bagian jalan dari badan jalan, termasuk bangunan pelengkap dan perlengkapannya yang tentu prasarana ini diperuntukkan bagi pengguna lalu lintas [1]. Sistem ini memainkan peran penting dalam meningkatkan keselamatan berkendara, terutama dalam mobil *self-driving* atau asisten pengemudi. Aspek utama dan terpenting dari mobil pribadi adalah kemampuan untuk menemukan jalan di antara berbagai fitur lainnya [2].

Kendaraan otonom, yang sering disebut sebagai mobil tanpa pengemudi atau mobil tanpa pengemudi, merupakan teknologi transformatif yang siap merevolusi sistem transportasi di seluruh dunia. Kendaraan ini telah menarik perhatian dan investasi yang signifikan dalam beberapa tahun terakhir karena potensinya untuk meningkatkan keselamatan jalan, mengurangi kemacetan lalu lintas, dan meningkatkan mobilitas bagi individu dari berbagai demografi [3]. Sistem harus mampu mengatasi berbagai kondisi jalan yang dinamis dan seringkali tidak ideal, seperti perubahan pencahayaan drastis, cuaca buruk, bayangan objek, serta marka jalan yang pudar atau terhalang. Di antara berbagai tugas persepsi, deteksi lajur jalan yang akurat dan real-time memainkan peran penting dalam memastikan navigasi kendaraan otonom yang aman. Lajur digunakan untuk memandu kendaraan dalam mempertahankan posisinya di jalan, mematuhi peraturan lalu lintas, dan menghindari tabrakan dengan kendaraan lain atau rintangan[4].

Untuk mengatasi tantangan tersebut, penelitian modern beralih ke metode deep learning, khususnya Convolutional Neural Network (CNN). CNN merupakan terobosan dalam pengolahan citra karena kemampuannya untuk belajar dan mengekstrak fitur-fitur penting dari data visual secara otomatis [5]. Banyak produsen mobil kini berfokus pada pengembangan sistem autopilot jalan raya yang bertujuan untuk mengurangi stres dan kelelahan pengemudi sekaligus menyediakan fitur keselamatan. Sistem bantuan pengemudi canggih (ADAS) saat ini dapat membantu menjaga mobil tetap berada di jalurnya dan mendeteksi kendaraan di depan. Namun, pengemudi manusia tetap bertanggung jawab atas rintangan atau insiden serius dengan selalu memegang kemudi[6].

Deep learning adalah bentuk pembelajaran mesin yang memungkinkan komputer untuk belajar dari pengalaman dan memahami dunia dalam bentuk hierarki konsep [7]. Dalam konteks klasifikasi jalur jalan, CNN dapat digunakan untuk mengekstraksi fitur-fitur penting dari *video dashcam*, sehingga mampu mengenali marka jalan[8]. Kemampuan inilah yang menjadikannya untuk memecahkan masalah deteksi lajur yang penuh variasi.

Pada penelitian ini, akan dikembangkan dan diuji sebuah sistem deteksi marka jalan berbasis CNN yang diterapkan pada video dashcam. Sistem ini

diimplementasikan dalam sebuah aplikasi web menggunakan framework Flask untuk menyediakan antarmuka yang mudah diakses oleh pengguna. Dengan menguji model CNN pada skenario jalan tol di dunia nyata, penelitian ini bertujuan untuk memberikan kontribusi praktis dalam pengembangan teknologi kendaraan otonom dan menjadi dasar bagi penelitian lebih lanjut di bidang navigasi cerdas.

2. TINJAUAN PUSTAKA

Berdasarkan penelitian yang dilakukan oleh Anil Kumar pada tahun 2024 dengan judul " AUTOMATIC ROAD LANE DETECTION USING CONVOLUTIONAL NEURAL NETWORK", hasil penelitian menunjukkan bahwa sistem ini memiliki akurasi tinggi dalam berbagai kondisi menantang, seperti cahaya rendah dan cuaca buruk. Sistem ini menunjukkan efektivitasnya dalam meningkatkan sistem transportasi cerdas dan kemampuan mengemudi otonom. Penelitian ini juga menyoroti potensi untuk pengembangan lebih lanjut dalam aplikasi visi komputer terkait infrastruktur jalan dan keselamatan kendaraan [9].

Penelitian yang dilakukan Gulbakhram pada tahun 2024 dengan judul "Real-Time Road Lane-Lines Detection using Mask-RCNN Approa" menunjukan Model yang dikembangkan menunjukkan akurasi 90-98% selama 100 epoch pembelajaran dan mampu beroperasi secara real-time, seperti yang ditunjukkan dalam demonstrasi video langsung [2].

Penelitian yang dilakukan Rustam A pada tahun 2023 dengan judul "Mask R-CNN Approach to Real-Time Lane Detection for Autonomous Vehicles" Hasil eksperimen menunjukkan bahwa pendekatan berbasis Mask R-CNN ini mencapai akurasi tinggi (95%-98%) setelah 100 epoch pembelajaran. Model ini juga menunjukkan kinerja yang lebih baik [4].

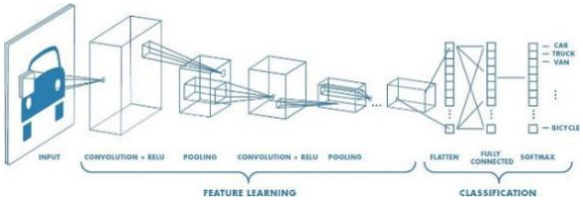
2.1. Deep Learning

Deep learning adalah cabang dari kecerdasan buatan (AI) yang menggunakan jaringan saraf tiruan dengan banyak lapisan (*deep neural networks*) untuk mempelajari representasi data secara otomatis dan hierarkis. Pendekatan ini sangat efektif dalam memproses data yang kompleks dan berstruktur tinggi seperti gambar, video, suara, dan teks.

Deep learning adalah bentuk pembelajaran mesin yang memungkinkan komputer belajar dari pengalaman dan memahami dunia dalam hierarki konsep. Buku ini menyajikan cakupan luas topik pembelajaran mendalam, mulai dari latar belakang matematis hingga teknik praktis yang digunakan di industri, serta aplikasi di berbagai bidang seperti visi komputer dan pemrosesan bahasa alami[10]. Dirancang untuk mahasiswa dan perangkat lunak, buku ini terbagi menjadi tiga bagian yang membahas dasar-dasar, algoritma mapan, dan ide-ide spekulatif untuk penelitian masa depan. Penulis berasumsi pembaca memiliki latar belakang ilmu komputer dasar, termasuk pemahaman tentang pemrograman dan kalkulus [7].

2.2. Convolusional Neural Network (CNN)

Convolutional neural network (CNN) merupakan salah satu algoritma bagian dari *deep learning*. CNN termasuk dalam jenis *deep neural network* karena kedalaman jaringannya yang tinggi dan banyak diaplikasikan pada data gambar. CNN mampu melakukan perhitungan matematis terhadap input yang terdiri dari banyak *hidden layer*. *Hidden layer* mengelola input gambar dan mengirimkan hasil olahan pada bagian *output*. CNN memiliki *convolutional layer* yang berguna untuk mengekstraksi fitur dan bagian *classification* untuk mengklasifikasikan fitur ke dalam kelas-kelas yang ditentukan dalam pelatihan. Tahap pelatihan membutuhkan waktu yang lebih lama untuk diselesaikan dan memerlukan perangkat komputasi yang mumpuni. Namun, proses prediksi CNN cukup cepat dan akurat [11].

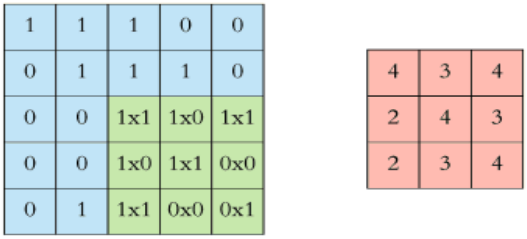


Gambar 1. Arsitektur CNN (Sumber: trivusi.web.id)

Arsitektur *Convolutional Neural Network* (CNN) pada Gambar 1 memproses data input, seperti gambar, melalui serangkaian lapisan. Dimulai dengan lapisan input, data kemudian melalui lapisan *Convolution* dan *ReLU* untuk mengekstrak fitur dan menambahkan *non-linearitas*. Lapisan *Pooling* selanjutnya mengurangi dimensi, diikuti oleh lapisan *Flatten* yang meratakan data menjadi vektor. Terakhir, lapisan *Classification* menggunakan fungsi seperti *softmax* untuk mengklasifikasikan data berdasarkan fitur yang telah dipelajari [12].

2.3. Convolution Layer

Di konvulusi ini dilaksanakan proses ekstraksi fitur pada citra dengan melakukan konvulusi antara kernel dengan sebuah citra. penggunaan banyak lapis layer dan kenel yang berbeda, maka akan diperoleh fitur di level tinggi seperti *edge*, *curve* dan fitur *color*[13]. Beberapa proses yang di butuhkan dalam konvulusi diantaranya sebagai berikut:



Gambar 2. Konvolusi dan Kernel (Sumber: trivusi.web.id)

$$O = \frac{(W - F + 2P)}{S} + 1 \quad \text{.....(1)}$$

Keterangan:

- W = dimensi input (tinggi atau lebar gambar),
- F = ukuran filter (misalnya, F=3),
- P = padding (biasanya P=1),
- S = stride (biasanya S=1).

Output dari konvolusi ini adalah fitur yang diekstraksi dari gambar input.

2.4. Relu

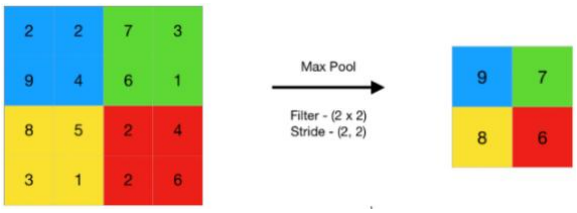
ReLU yaitu proses memperkenalkan non-linearitas dan meningkatkan representasi model. Nilai ouput neuron bernilai 0 jika nilai input negatif. Jika neutron tersebut bernilai input positif adalah nilai input trigger itu sendiri. *ReLu* juga sering kali digunakan karena dapat melatih neural network lebih cepat [14].

$$ReLU(x) = max(0, x) \quad \text{.....(2)}$$

Dengan kata lain, setiap nilai negatif dari output konvolusi diubah menjadi nol, sementara nilai positif tetap sama.

2.5. Pooling Layer

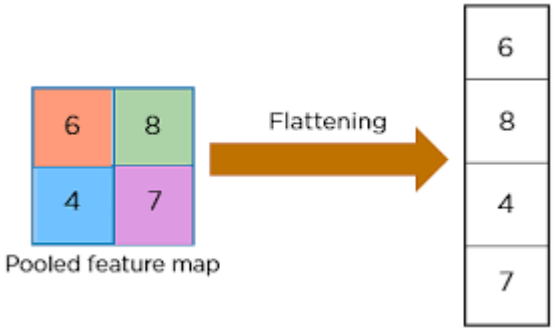
Pooling untuk mengurangi dimensi dari hasil konvolusi. Biasanya, pooling menggunakan operasi max pooling dengan filter berukuran 2×2 dan stride 2.



Gambar 3. *Max pooling* (Sumber: trivusi.web.id)

2.6. Flatten Layer

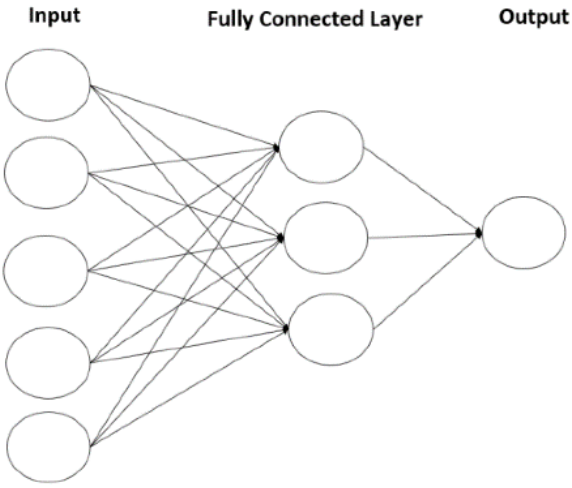
Flatten adalah menjadikan satu vektor satu dimensi untuk diteruskan ke lapisan *fully connected*.



Gambar 4. *Flatten* (Sumber: trivusi.web.id)

2.7. Fully Connected Layer

Pada lapisan *fully connected*, vektor hasil *flatten* diteruskan ke layer yang terhubung sepenuhnya dengan neuron lainnya. Mengambil fitur-fitur yang telah diekstraksi dan diratakan, lalu menggunakannya untuk melakukan inferensi atau klasifikasi berdasarkan kombinasi non-linear dari fitur-fitur tersebut.



Gambar 5. Fully Connected Layer (Sumber: trivusi.web.id)

2.8. Softmax

Untuk klasifikasi, output dari lapisan *fully connected* diteruskan ke fungsi *softmax* yang akan menghasilkan probabilitas untuk setiap kelas.

$$F_j = \frac{e^{zj}}{\sum_k e^{zk}} \quad \text{.....(3)}$$

- z_j = output dari neuron j,
- e^{zj} = eksponensial dari nilai output neuron j,
- $\sum_k e^{zk}$ = jumlah eksponensial dari output seluruh kelas.

Probabilitas ini akan menunjukkan seberapa besar kemungkinan gambar tersebut termasuk dalam kelas tertentu. Dalam kasus ini, p0 dan p1 adalah probabilitas untuk dua kelas yang berbeda, seperti "marka jalan terdeteksi" dan "marka jalan tidak terdeteksi".

2.9. OpenCV

Dalam penerapannya, *OpenCV* sering digunakan dalam berbagai bidang, seperti sistem pengawasan, augmented reality, deteksi kendaraan, dan pengolahan gambar medis. Dengan kombinasi algoritma pemrosesan gambar dan *deep learning*, *OpenCV* mampu mendukung pengembangan sistem cerdas seperti pengenalan objek dalam video secara *real-time*. Selain itu, *OpenCV* juga kompatibel dengan berbagai pustaka lain, seperti *TensorFlow* dan *PyTorch*, sehingga dapat diintegrasikan dengan model *deep learning*. Dengan fitur yang lengkap dan kemudahan penggunaannya, *OpenCV* menjadi salah satu alat utama dalam dunia visi komputer modern [15].

3. METODE PENELITIAN

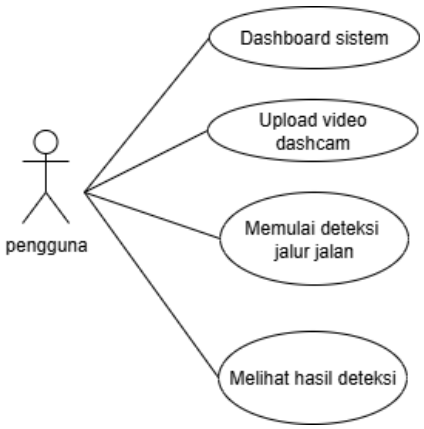
3.1. Pengambilan Data

Pengambilan data video untuk penelitian ini dilakukan pada tanggal 10 dan 12 Juli 2025. Proses ini bertujuan untuk mendapatkan rekaman video *dashcam* pada rute yang telah ditentukan Tol Singosari - Pakis. Pada hari pertama, pengambilan data pada pagi hari dengan kondisi cuaca cerah dan pada siang hari dengan

kondisi cuaca cerah. Pada tanggal 12 Juli, dilanjutkan pada pagi hari dengan kondisi cuaca cerah.

3.2. Use Case Diagram

Dalam hal ini, aktornya adalah pengguna yang akan dijalankan oleh sistem. Berikut ini adalah *use case* diagram dapat dilihat pada gambar 6.



Gambar 6. Use case diagram

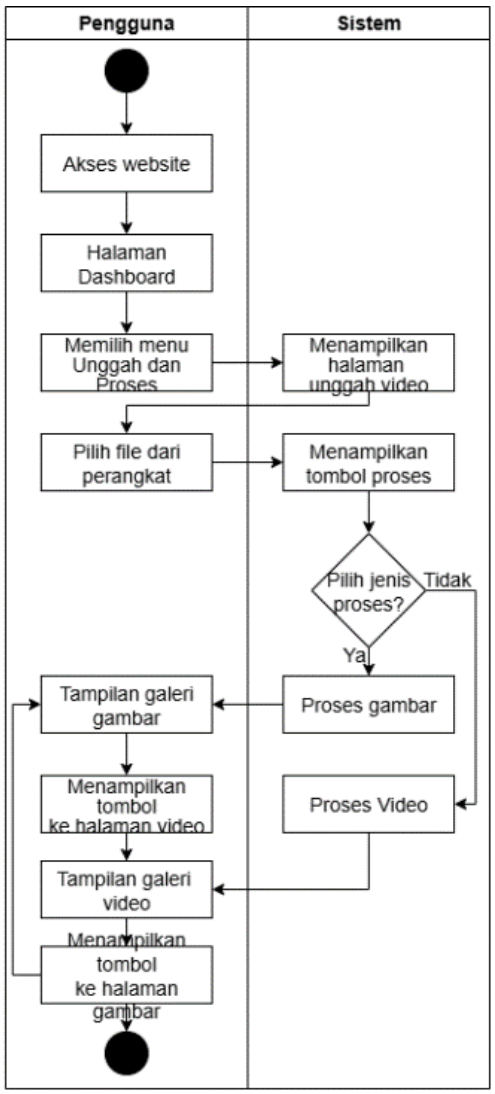
Pada gambar 6. menggambarkan pengguna sebagai aktor utama yang berinteraksi dengan antarmuka *website*. Pengguna pertama kali memasuki Halaman *Dashboard* yang berfungsi untuk pengguna mengetahu cara kerja sistem yang digunakan. Selanjutnya pada Upload video untuk memulai proses analisis. Dari halaman ini, pengguna dapat mengakses fitur Proses Deteksi, di mana sistem akan melakukan pemrosesan video menggunakan *computer vision* untuk mengidentifikasi objek atau pola tertentu.

Setelah proses selesai, hasilnya ditampilkan di bagian Hasil Deteksi, yang menyajikan informasi visual berhasil diidentifikasi oleh sistem. Terakhir, pengguna dapat mengakses Laporan Deteksi untuk melihat rekapan hasil analisis dalam bentuk video dan gambar per *frame*.

3.3. Activity Diagram

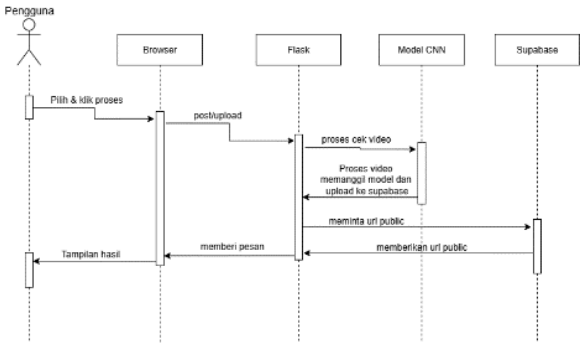
Pada gambar 7 adalah alur kerja utama dalam aplikasi deteksi marka jalan, mulai dari pengguna membuka aplikasi hingga menampilkan hasil proses deteksi. Diagram ini bertujuan untuk memvisualisasikan interaksi antara pengguna dan sistem. Sistem memberikan pilihan bagi pengguna untuk memilih hasil deteksi akan diproses dan ditampilkan. Pengguna dapat memilih hasil dalam bentuk gambar per *frame*, yang akan ditampilkan ke galeri, atau sebagai video, yang langsung dapat diputar.

Pada gambar 8 diagram alur komunikasi antar komponen dalam sistem deteksi marka jalan berbasis web, yang melibatkan pengguna, browser, Flask, model CNN untuk pemrosesan video, dan Supabase sebagai penyimpanan *cloud*. Proses dimulai saat pengguna memilih dan mengunggah *file* video melalui browser, yang kemudian dikirim ke server Flask melalui permintaan POST. Server akan memverifikasi *file*, memproses video menggunakan model CNN, dan mengunggah hasilnya ke *Supabase*. Setelah itu, *Flask* meminta URL publik dari *Supabase* untuk *file* yang telah diunggah, lalu mengembalikan URL tersebut ke browser agar dapat ditampilkan kepada pengguna.



Gambar 7. Activity Diagram Pengguna

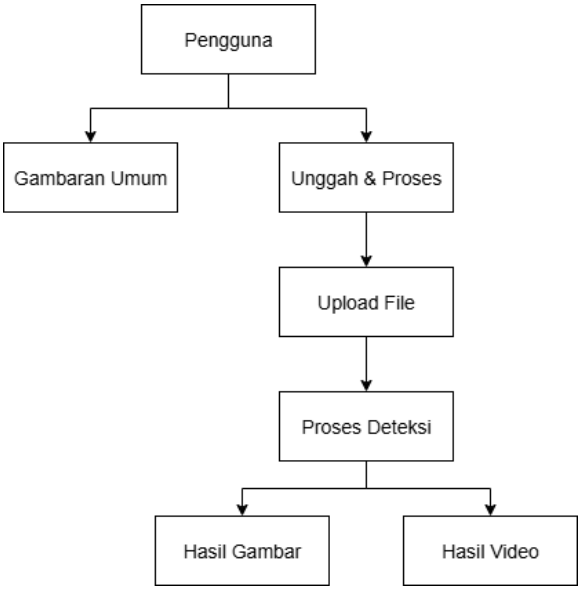
Sequence Diagram



Gambar 8. Sequence Diagram Pengguna

3.4. Struktur Menu

Pada gambar 9 merupakan struktur menu *website* yang akan dibuat, ada beberapa menu yang dibutuhkan sebagai berikut:



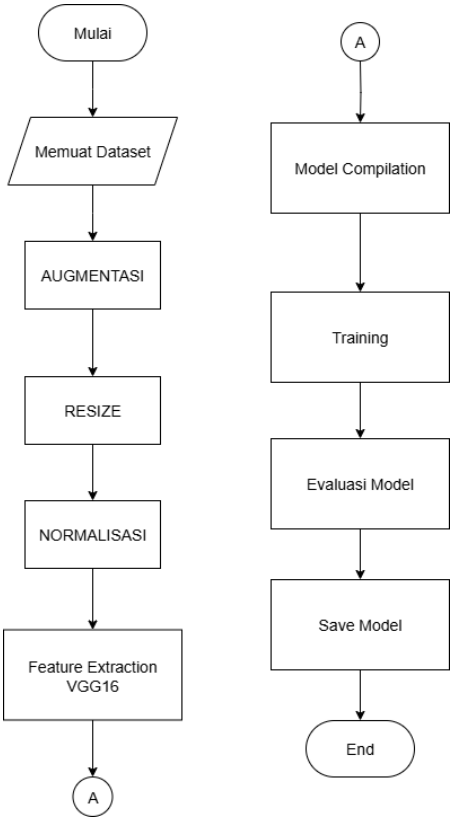
Gambar 9. Struktur Menu *Website* Deteksi Jalur Jalan

Pada gambar 9 merupakan struktur menu yang yang dirancang untuk pengguna dalam sistem deteksi jalur jalan. Menu utama terdiri dari *Dashboard*, yang berfungsi untuk memberikan pemahaman bagi pengguna tentang cara kerja *website*. Halaman Unggah dan proses tempat pengguna dapat memulai proses deteksi jalur. Pengguna akan diarahkan untuk *upload file* berupa video. Proses Deteksi yang menampilkan langkah-langkah deteksi. Setelah proses selesai, pengguna dapat melihat Hasil Gambar dan Hasil Video

3.5. Flowchart CNN

Pada gambar 10 adalah alur kerja metode CNN (*Convolutional Neural Network*) menggunakan arsitektur *VGG16* untuk pemrosesan data gambar, mulai dari tahap persiapan hingga penyimpanan model. Proses diawali dengan memuat dataset gambar yang akan digunakan. Selanjutnya, data tersebut melalui tahap *pre-processing* yang meliputi augmentasi untuk meningkatkan variasi data, *resize* untuk menyeragamkan ukuran gambar, dan normalisasi agar nilai piksel berada dalam rentang yang sesuai.

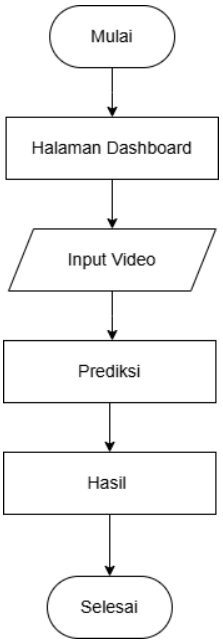
Setelah *pre-processing*, fitur-fitur gambar diekstraksi menggunakan model *VGG16* yang telah dilatih sebelumnya. Model kemudian dikompilasi dengan menambahkan layer *fully connected* dan mengkonfigurasi *optimizer*, *loss function*, serta metrik evaluasi. Tahap pelatihan (*training*) dilakukan untuk melatih model mengenali pola dari data gambar. Setelah pelatihan selesai, model dievaluasi menggunakan data validasi atau *testing* untuk memastikan performanya optimal dan tidak overfitting. Model disimpan dalam format file tertentu agar dapat digunakan kembali tanpa perlu pelatihan ulang. Proses ini berakhir setelah model berhasil disimpan, menandakan seluruh alur kerja telah selesai.



Gambar 10. *Flowchart CNN*

3.6. Flowchart Sistem

Gambar 11 adalah gambaran proses sistem aplikasi yang akan dibuat, berikut gambar *flowchart* sistem.



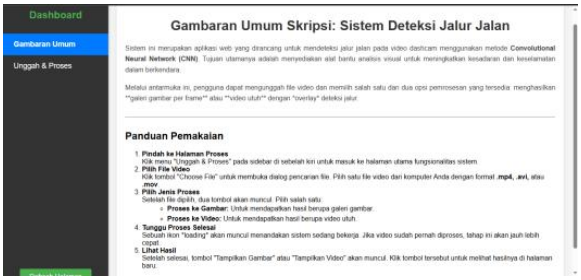
Gambar 11. *Flowchart* sistem

Pada gambar 11 alur kerja utama dari sistem deteksi jalur jalan. Proses dimulai dari pengguna diarahkan ke Halaman *Dashboard* sebagai antarmuka utama. Pengguna dapat mengunggah *Input* Video ke sistem untuk diproses. Setelah video dimasukkan, sistem melakukan Prediksi, di mana algoritma CNN menganalisis video tersebut untuk menghasilkan *output* deteksi jalur jalan. Hasil prediksi kemudian ditampilkan kepada pengguna dalam bagian Hasil, yang bisa berupa video deteksinya dan gambar per *frame*. Flowchart ini menunjukkan alur sederhana yang memandu pengguna dalam menggunakan sistem deteksi marka jalan.

4. HASIL DAN PEMBAHASAN

Pada hasil dan pembahasan ini, menampilkan antarmuka *website* yang telah dibuat sistem untuk deteksi marka jalan menggunakan *framework flask*. Sistem memiliki halaman *dashboard* dan halaman hasil.

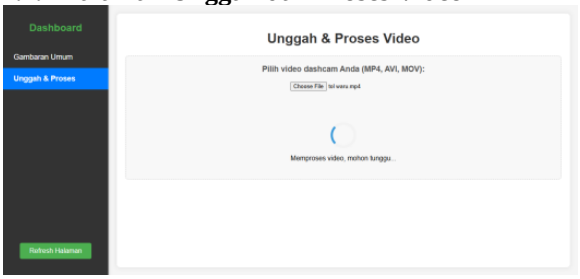
4.1. Halaman Dashboard



Gambar 12. Halaman *dashboard*

Pada gambar 12 adalah halaman *dashboard* yang dirancang terdiri dari *sidebar* navigasi dan area konten utama. Pada *sidebar* navigasi menyediakan "Gambaran Umum" dan "Unggah & Proses". Sementara itu, area konten utama berfungsi sebagai pusat informasi, memberikan informasi singkat mengenai tujuan sistem dan memberikan panduan pemakaian yang memandu pengguna langkah-langkah alur sistem *website*.

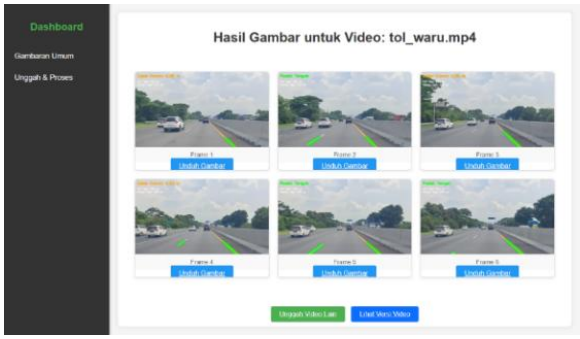
4.2. Halaman Unggah dan Proses Video



Gambar 13. Halaman Unggah Dan Proses Video

Pada gambar 13 menampilkan halaman unggah dan proses video yang terdapat “Pilih video dashcam Anda" yang memandu pengguna tentang langkah pertama yang perlu dilakukan. Selanjutnya antarmuka menampilkan tombol "*Choose File*", yang memungkinkan pengguna memilih file video dari perangkat mereka. Setelah video dipilih, pengguna dapat mengklik tombol "Proses" untuk memulai proses dan menampilkan dua tombol yaitu tampilkan gambar dan tampilkan video.

4.3. Halaman Gambar



Gambar 14. Halaman Gambar

Pada gambar 14 adalah halaman gambar yang menampilkan hasil dari pengguna memilik memproses video dengan gambar. Pada halaman tersebut hasil deteksi sistem gambar yang sudah rapi. Pengguna bisa mengunduh gambar tersebut. Pada bagian bawah terdapat dua tombol, tombol warna hijau untuk mengupload *file* lagi dan tombol berwarna biru untuk tampilkan video.

4.4. Halaman Video

Gambar 15. Halaman Video

Pada gambae 15 merupakan halaman video yang berfungsi sebagai streaming hasil deteksi. Pengguna dapat melihat secara streaming dan juga bisa di unduh.

4.5. Pengujian Fungsional

Pengujian sistem dilakukan untuk memverifikasi mekanisme validasi input pada fitur unggah video. Pengujian ini mencakup dua skenario utama yang bertujuan untuk memastikan sistem dapat menangani input yang tidak valid dengan tepat.

Tabel 1. Pengujian Fungsional

Aktivitas Pengujian	Pengujian yang dilakukan	Hasil Yang di Harapkan	Hasil Pengujian	Kesimpulan
Menguji Proses Semua	Upload File Kosong	Proses ditolak dan menampilkan *pop up* "Peringatan! Anda belum memilih file video. Silakan pilih file sebelum mengunggah.*"*	Proses ditolak dan menampilkan *pop up* "Peringatan! Anda belum memilih file video. Silakan pilih file sebelum mengunggah.*"*	Berhasil
	Uploud File Berformat Bukan video	Proses ditolak dan menampilkan *pop up* "*Error* Validasi! Format file tidak didukung! Hanya MP4, AVI, MOV yang diizinkan.*"*	Proses ditolak dan menampilkan *pop up* "*Error* Validasi! Format file tidak didukung! Hanya MP4, AVI, MOV yang diizinkan.*"*	Berhasil
	Upload File berformat mp4	Menampilkan tombol vidio dan gambar muncul	Menampilkan tombol vidio dan gambar muncul	Berhasil

Tabel 2 menunjukan pengujian fungsional bahwa validasi *input* pada sistem berfungsi dengan baik. Sistem berhasil menolak pengiriman *form* kosong dan upaya mengunggah format file yang tidak didukung (seperti PDF atau JPG). Dalam kedua skenario tersebut, sistem memberikan pesan peringatan dan *error* yang jelas serta informatif kepada pengguna. Proses unggah untuk format video yang valid (MP4, AVI, MOV) juga berjalan dengan sukses sesuai harapan.

4.6. Pengujian metode

Tabel 1 menggunakan data 10 video uji coba yang digunakan. Video ini di ambil pada tanggal 17 Juli 2025 yang dimulai dari masuk tol Singosari keluar tol Pakis.

Tabel 2. Pengujian metode

No	Video	Tanggal	Kondisi Cuaca	Jalan	
				Lurus	Belokan
1.	Tol-Singosari-1.mp4	17-07-2025	Cerah	✓	
2.	Tol-Singosari-2.mp4	17-07-2025	Cerah	✓	✓
3.	Tol-Singosari-3.mp4	17-07-2025	Cerah	✓	
4.	Tol-Singosari-4.mp4	17-07-2025	Cerah	✓	✓
5.	Tol-Singosari-5.mp4	17-07-2025	Cerah		✓
6.	Tol-Singosari-6.mp4	17-07-2025	Cerah	✓	
7.	Tol-Singosari-7.mp4	17-07-2025	Cerah	✓	
8.	Tol-Singosari-8.mp4	17-07-2025	Cerah	✓	✓
9.	Tol-Singosari-9.mp4	17-07-2025	Berawan	✓	✓
10.	Tol-Singosari-10.mp4	17-07-2025	Cerah		✓

Dari tabel 2, Pengujian kinerja model dilakukan menggunakan 10 video uji yang karakteristiknya, seperti kondisi cuaca dan tipe jalan, didokumentasikan untuk menganalisis faktor-faktor yang memengaruhi hasil deteksi. Mayoritas video diambil pada kondisi ideal (cuaca cerah, jalan lurus), namun secara sengaja disertakan juga variasi penting seperti jalan berbelok untuk menguji ketahanan model.

Variasi jalan berbelok ini menjadi faktor penguji, karena saat menikung, perspektif marka jalan menjadi sangat melengkung dari sudut pandang kamera. Pola visual yang berbeda signifikan dari jalan lurus ini menjadi tantangan bagi model CNN, dan dianalisis sebagai penyebab utama mengapa hasil klasifikasi cenderung menjadi kurang stabil atau hanya menghasilkan "Deteksi Sebagian" pada frame-frame yang menunjukkan belokan.

Pada tabel 3 menunjukkan informasi proses deteksi marka jalan yang dilakukan pada sepuluh video uji. Setiap video memiliki jumlah *frame per second* (FPS), durasi video per detik, dan total waktu proses yang dibutuhkan untuk menyelesaikan seluruh frame.

Tabel 3.Pengujian waktu komputasi

No	Video	Frame Per Second	Durasi Video (Detik)	Waktu Proses (Detik)
1.	Tol-Singosari-1.mp4	41	60	121
2.	Tol-Singosari-2.mp4	40	60	135
3.	Tol-Singosari-3.mp4	40	60	118
4.	Tol-Singosari-4.mp4	40	60	129
5.	Tol-Singosari-5.mp4	40	60	115
6.	Tol-Singosari-6.mp4	41	60	132
7.	Tol-Singosari-7.mp4	40	60	138
8.	Tol-Singosari-8.mp4	40	60	142
9.	Tol-Singosari-9.mp4	40	60	125
10.	Tol-Singosari-10.mp4	16	23	48

Berdasarkan tabel 3, untuk mengukur efisiensi komputasi sistem. Hasil pengujian menunjukkan korelasi linear antara durasi video dengan waktu pemrosesan. Untuk video berdurasi 60 detik, waktu proses rata-rata yang dibutuhkan adalah 128.3 detik.

Untuk mengkontekstualisasikan apakah waktu ini "cepat" atau "lambat", kita dapat mengubahnya ke metrik standar dalam computer vision, yaitu *Frame per Second* (FPS), yang mengukur berapa banyak *frame* yang dapat diproses oleh sistem setiap detiknya. Dengan asumsi video uji berjalan pada 30 FPS, maka video 60 detik memiliki total 1800 frame.

Menurut penelitian G. Beissenova [2] dalam bidang deteksi jalur untuk kendaraan otonomsecara Real Time, sebuah sistem dianggap dapat berjalan secara real-time jika mampu memproses video dengan kecepatan yang setara atau melebihi kecepatan frame kamera, yaitu sekitar 25-30 FPS.

5. KESIMPULAN DAN SARAN

Penelitian ini mengembangkan dan menerapkan sistem deteksi marka jalan berbasis video dashcam menggunakan metode *Convolutional Neural Network* (CNN). Model menunjukkan performa deteksi yang baik pada kondisi marka yang jelas dan jalan lurus, namun masih kurang stabil saat menghadapi kondisi jalan berbelok, marka yang samar, atau cuaca yang kurang ideal. Dari pengujian terhadap 10 video berdurasi 60 detik dengan FPS rata-rata 40–41, waktu pemrosesan berkisar antara 115–142 detik, sedangkan video berdurasi 23 detik (FPS 16) hanya membutuhkan 48 detik. Untuk pengembangan selanjutnya, disarankan agar dilakukan perbandingan metode dengan pendekatan *deep learning* lain di luar CNN serta memperluas variasi dataset pelatihan yang tetap relevan dengan skenario deteksi jalur jalan agar sistem dapat beradaptasi lebih baik terhadap kondisi jalan yang beragam.

DAFTAR PUSTAKA

[1] E. Setiadi and A. Wibowo, "Klasifikasi dan Deteksi Keretakan Pada Trotoar Menggunakan Metode Convolutional Neural Network," 2023.

[2] G. Beissenova *et al.*, "Real-Time Road Lane-Lines Detection using Mask-RCNN Approach," 2024. [Online]. Available: www.ijacsa.thesai.org

[3] A. Bhandari, S. Sakore, and S. Kale, "Detection of Lane & Speed Breakers for Autonomous Vehicles using CNN," *Int J Res Appl Sci Eng Technol*, vol. 12, no. 4, pp. 5725–5729, Apr. 2024, doi: 10.22214/ijraset.2024.61362.

[4] R. Abdrakhmanov *et al.*, "Mask R-CNN Approach to Real-Time Lane Detection for Autonomous Vehicles," 2023. [Online]. Available: www.ijacsa.thesai.org

[5] I. Maulana, N. Khairunisa, R. Mufidah Informatika, U. H. Singaperbangsa Karawang Jl Ronggo Waluyo, T. Timur, and J. Barat, "DETEKSI BENTUK WAJAH MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK (CNN)," 2023.

[6] Monowar Hossain Saikat, Sonjoy Paul Avi, Kazi Toriqul Islam, Tanjida Tahmina, Md Shahriar Abdullah, and Touhid Imam, "Real-Time Vehicle and Lane Detection using Modified OverFeat CNN: A Comprehensive Study on Robustness and Performance in Autonomous Driving," *Journal of Computer Science and Technology Studies*, vol. 6, no. 2, pp. 30–36, Apr. 2024, doi: 10.32996/jcsts.2024.6.2.4.

[7] K. G. Kim, "Book Review: Deep Learning," *Healthc Inform Res*, vol. 22, no. 4, p. 351, 2016, doi: 10.4258/hir.2016.22.4.351.

[8] A. Ridho, A. F. Setiawan, and N. Vendyansyah, "KLASIFIKASI KUALITAS KAYU DENGAN METODE K-NEAREST NEIGHBORS (KNN) BERBASIS WEBSITE," 2024.

[9] A. K. Singh, A. Kumar, A. Singh, and M. Chandra, "AUTOMATIC ROAD LANE DETECTION USING CONVOLUTIONAL NEURAL NETWORK," Innovative Research Publication, Apr. 2024, pp. 57–61. doi: 10.55524/csistw.2024.12.1.10.

[10] C. W. Wiguna, J. Dedy Irawan, and M. Orisa, "PENERAPAN METODE CONVOLUTIONAL NEURAL NETWORK PADA APLIKASI DETEKSI WAJAH BURONAN BERBASIS WEB," 2022.

[11] F. Hafifah, S. Rahman, and S. Asih, "Klasifikasi Jenis Kendaraan Pada Jalan Raya Menggunakan Metode Convolutional Neural Networks (CNN)," vol. 2, no. 5, pp. 292–301, 2021, [Online]. Available: https://ejurnal.seminar-id.com/index.php/tin

[12] A. Prayoga, Maimunah, P. Sukmasetya, Muhammad Resa Arif Yudianto, and Rofi Abul Hasani, "Arsitektur Convolutional Neural Network untuk Model Klasifikasi Citra Batik Yogyakarta," *Journal of Applied Computer Science and Technology*, vol. 4, no. 2, pp. 82–89, Nov. 2023, doi: 10.52158/jacost.v4i2.486.

[13] R. O. Yumame, Y. A. Pranoto, and J. Dedy Irawan, "RANCANG BANGUN PENDETEKSI MASKER DAN ALAT PEMBERSIH TANGAN OTOMATIS BERBASIS ARDUINO MENGGUNAKAN METODE CNN (CONVULUTION NEURAL NETWORK) DI GEREJA GKI JAYAPURA PAPUA," 2022.

[14] Hilman Fatoni, R. Diva A M, and Adam, "DETEKSI DAN KLASIFIKASI KENDARAAN RINGAN DAN BERAT UNTUK JALAN TOL MENGGUNAKAN CNN," 2024.

[15] Lia Farokhah, "Perbandingan Metode Deteksi Wajah Menggunakan OpenCV Haar Cascade, OpenCV Single Shot Multibox Detector (SSD) dan DLib CNN," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 5, no. 3, pp. 609–614, Jun. 2021, doi: 10.29207/resti.v5i3.3125.

[16] G. Beissenova *et al.*, "Real-Time Road Lane-Lines Detection using Mask-RCNN Approach," 2024. [Online]. Available: www.ijacsa.thesai.org