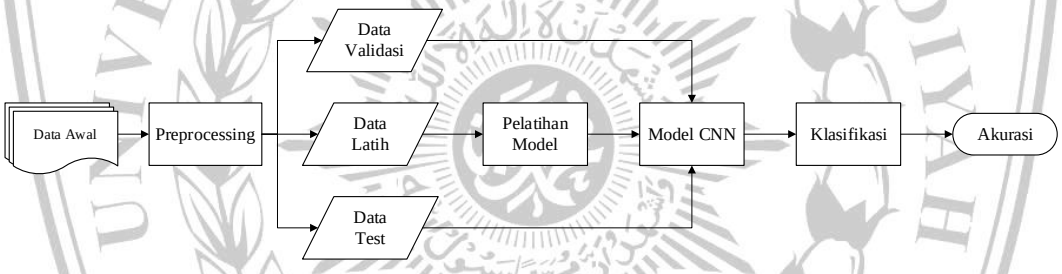
BAB III

METODE PENELITIAN

Penelitian ini memiliki tujuan utama untuk mengembangkan sistem yang dapat secara otomatis mengenali dan mengklasifikasi kerusakan pada bodi mobil. Untuk mencapai tujuan ini, memilih metode *Convolutional Neural Network* (CNN), yang telah terbukti sangat efektif dalam tugas pengenalan gambar.

Gambar 3.1 yang terlihat di bawah ini adalah representasi metode penelitian yang akan digunakan. Model ini telah dirancang dengan hati-hati dan diatur untuk mencapai tingkat akurasi yang tinggi dalam mengklasifikasi kerusakan pada bodi mobil. Melalui penelitian ini, harapannya dapat memberikan kontribusi penting dalam pengembangan teknologi deteksi kerusakan otomatis yang bermanfaat dalam industri perawatan dan perbaikan kendaraan.



Gambar 3.1 Metode Penelitian

Diagram pada gambar 3.1 di atas menunjukkan aliran data dari sumber data ke unit pemrosesan data. Sumber data adalah data yang sedang diproses. Unit pemrosesan data terdiri dari dua tahap, yaitu tahap *pre-processing* dan tahap pelatihan.

Pada tahap *pre-processing*, data dibersihkan dan diubah menjadi format yang dapat diproses oleh unit pelatihan. Tahap ini meliputi langkah-langkah seperti penghapusan *noise* (*gaussian blur*), pengubahan ukuran, dan penajaman gambar. Pada tahap pelatihan, model konektivitas saraf tiruan (CNN) dilatih untuk mengenali pola dalam data. Proses pelatihan ini dilakukan dengan menggunakan algoritma pembelajaran mesin.

Setelah model CNN dilatih, model tersebut dapat digunakan untuk mengklasifikasikan data baru. Proses klasifikasi ini dilakukan dengan menggunakan algoritma klasifikasi.

Akurasi model CNN dapat diukur dengan menggunakan data validasi. Data validasi adalah data yang tidak digunakan untuk pelatihan, tetapi digunakan untuk mengukur kinerja model.

3.1 Pengumpulan Data

Metode CNN adalah pendekatan yang sangat tepat untuk tugas ini karena mampu mengambil fitur-fitur penting dari gambar mobil dan memahami pola kerusakan yang mungkin ada. Dataset penelitian ini didapatkan dari website Kaggle [24]. Dimana data keseluruhan berjumlah 2300 data. Dataset ini mencakup berbagai gambar mobil yang mengalami berbagai jenis kerusakan, termasuk goresan, penyok, kaca pecah, dan kerusakan struktural akibat kecelakaan. Hal ini memungkinkan untuk melatih model dengan beragam data sehingga model dapat lebih baik dalam mengenali dan mengklasifikasi berbagai jenis kerusakan. Selanjutnya, data akan dibagi menjadi dua kategori, yaitu ‘tidak rusak’ dan ‘rusak’.

Gambar 3.2 Mobil Rusak

Gambar 3.3 Mobil Tidak Rusak

3.2 *Pre-processing*

Pre-processing adalah tahapan untuk menghilangkan beberapa permasalahan yang bisa mengganggu saat pemrosesan data. Ini dilakukan melalui beberapa langkah, yang diharapkan dapat menghasilkan hasil akhir yang sebaik mungkin. Langkah-langkah yang diperlukan adalah:

3.2.1 *Sharpening Images* dan *Random Contras*

Teknik ini digunakan untuk meningkatkan tajam dan detail gambar dengan menonjolkan tepi. Penajaman gambar biasanya digunakan dalam industri percetakan dan fotografi untuk meningkatkan kontras lokal serta mempertajam gambar.

3.2.2 *Resize images*

Dalam langkah ini, untuk memvisualisasikan perubahan, kita akan membuat dua fungsi yang bertujuan untuk menampilkan gambar. Fungsi pertama digunakan untuk menampilkan satu gambar, sementara fungsi kedua digunakan untuk menampilkan dua gambar. Setelah itu, kita akan membuat fungsi pemrosesan yang hanya menerima gambar sebagai parameter. Karena beberapa gambar yang diambil oleh kamera bervariasi dalam ukuran, kita perlu menetapkan ukuran dasar untuk semua gambar yang akan diumpankan ke algoritma AI.

3.2.3 Menghilangkan *Noise* (*Denoise*).

Di dalam fungsi Processing, tambahkan langkah ini untuk menghaluskan gambar dan menghilangkan *noise* yang tidak diinginkan. Langkah ini menggunakan *Gaussian blur*.

Gaussian blur, juga dikenal sebagai *Gaussian smoothing*, adalah hasil pengaburan gambar menggunakan fungsi *Gaussian*. Efek ini sering digunakan dalam perangkat lunak grafis dan biasanya untuk mengurangi *noise* pada gambar. Efek visual dari teknik blur ini adalah gambar yang halus yang tampak disinari cahaya, mirip dengan tampilan gambar melalui layar. Ini sangat berbeda dengan efek bokeh yang dihasilkan oleh lensa yang tidak fokus atau bayangan suatu objek dalam pencahayaan normal. Pemulusan *Gaussian* juga digunakan sebagai langkah pra-pemrosesan dalam algoritma visi komputer untuk meningkatkan struktur gambar pada skala yang berbeda. Untuk memulai, kita memerlukan fungsi *Gaussian* dalam dua dimensi:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (3.1)$$

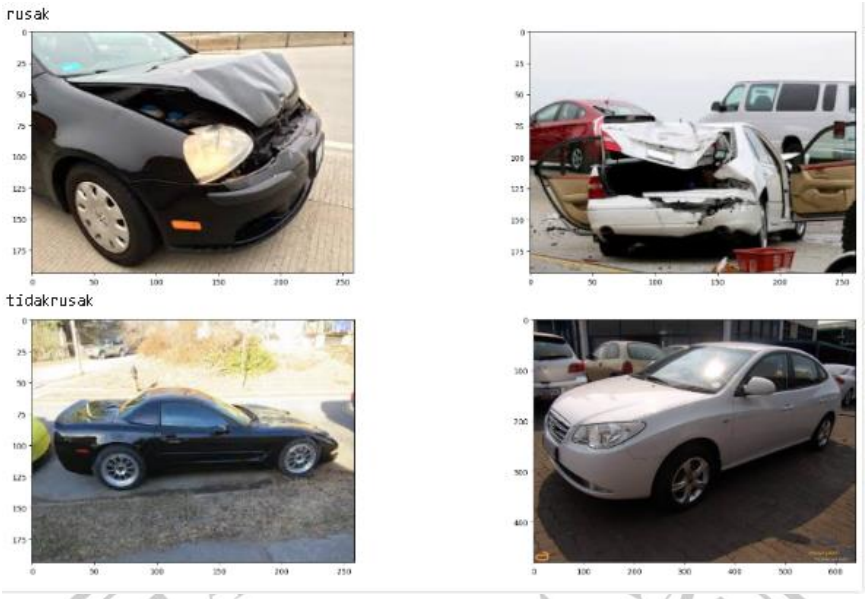
Nilai dari fungsi ini akan menciptakan matriks/ kernel konvolusi yang diterapkan pada setiap piksel dalam gambar asli. Kernel biasanya sangat kecil. Semakin besar ukuran kernel, semakin banyak perhitungan yang harus dilakukan untuk setiap piksel.

Nilai x dan y menentukan perubahan delta pada piksel tengah (0, 0). Misalnya, jika radius yang dipilih untuk kernel adalah 3, nilai x dan y akan berkisar dari -3 hingga 3 (inklusif).

Simpangan baku σ mempengaruhi seberapa signifikan pengaruh piksel-piksel tetangga terhadap hasil komputasi pada piksel tengah. Secara teknis, dalam fungsi *Gaussian*, karena fungsi ini meluas tanpa batas, Anda dapat berargumen bahwa Anda perlu mempertimbangkan setiap piksel dalam gambar untuk mendapatkan efek buram yang "benar". Namun, dalam praktiknya, piksel di luar 3σ memiliki dampak yang sangat kecil pada nilai yang dihasilkan.

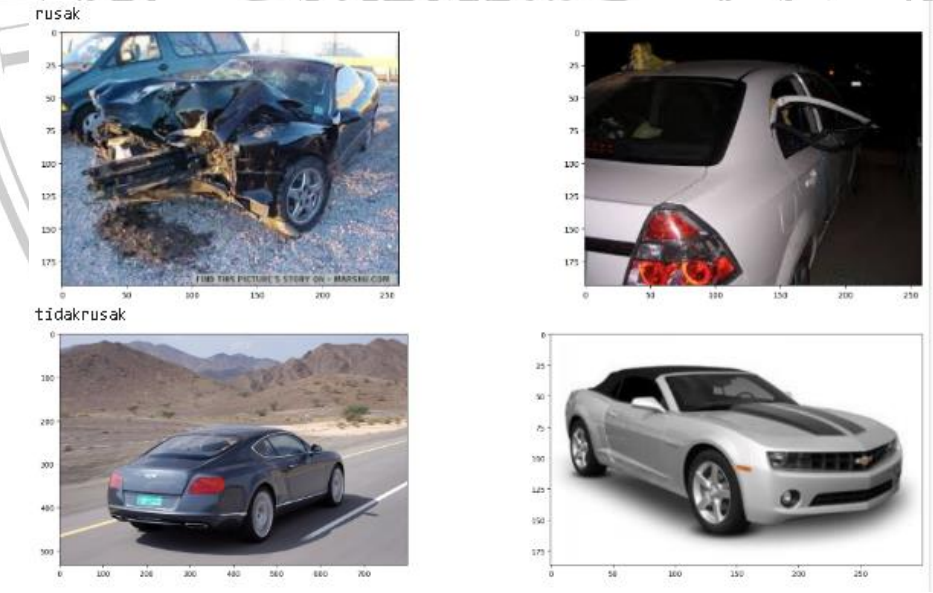
3.3 Pembagian Data

Data yang diperoleh akan dibagi menjadi data pelatihan, data validasi, dan data uji dengan perbandingan 70% untuk data pelatihan, 20% untuk data validasi, dan 10% untuk data uji. Sebelumnya, telah dijelaskan bahwa total data yang tersedia adalah 2300. Kemudian, data akan dibagi menjadi 1610 data uji, di mana setiap kelas terdiri dari 805 data gambar untuk pengujian, 460 data untuk validasi (dengan 230 data per kelas), dan 230 data untuk pengujian (dengan 115 data per kelas). Penggunaan dataset ini pada metode CNN penting untuk melatih model agar dapat membedakan antara mobil yang rusak dan tidak rusak dengan baik



Gambar 3.4 Contoh Data *Training*

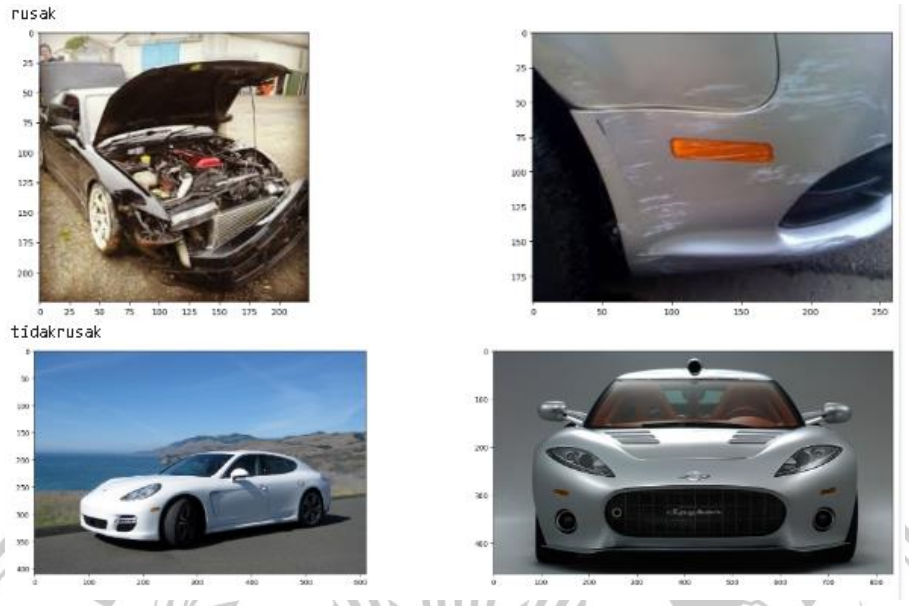
Pada Gambar 3.4, terdapat ilustrasi contoh data *training* yang menampilkan sampel data yang akan digunakan untuk melatih model *Convolutional Neural Network* (CNN). Data *training* ini memuat sejumlah contoh yang akan dijadikan acuan oleh model CNN dalam proses pembelajaran.



Gambar 3.5 Contoh Data *Validasi*

Gambar 3.5 menunjukkan contoh data validasi yang digunakan untuk melakukan verifikasi pada model *Convolutional Neural Network* (CNN). Data

validasi ini berperan penting dalam menguji kinerja model CNN terhadap dataset yang tidak digunakan selama pelatihan.



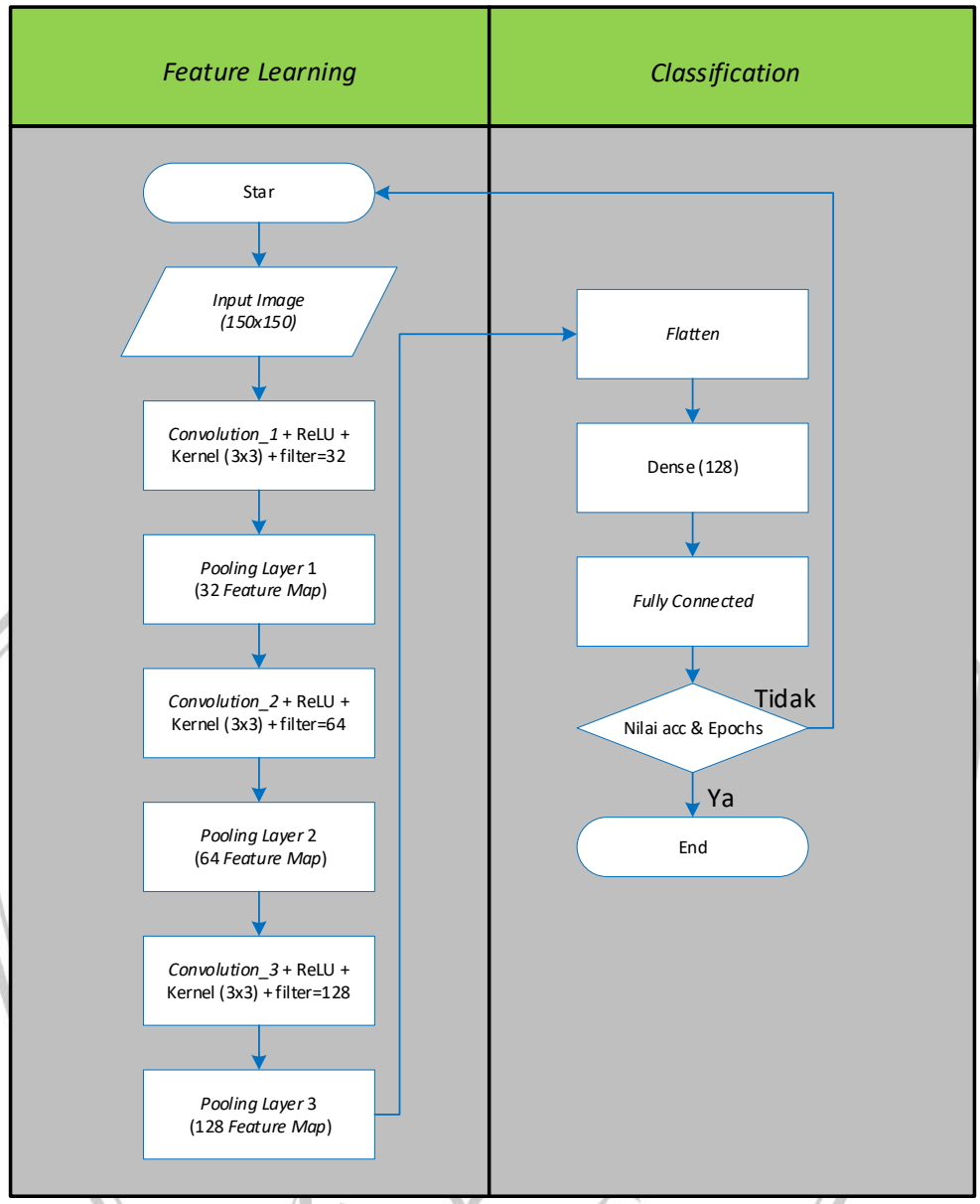
Gambar 3.6 Contoh Data *Testing*

Gambar 3.6 menampilkan Contoh Data *Testing* yang digunakan untuk menguji model yang telah dilatih. Data tersebut adalah contoh representatif yang akan diujikan pada model guna mengukur kinerjanya setelah melalui proses pelatihan.

3.4 Rancangan *Convolutional Neural Network*s (CNN)

Setelah data dibuat, langkah selanjutnya adalah melatih model CNN. Secara umum, CNN memiliki dua tahap, yaitu tahap *Feature Learning* dan tahap klasifikasi. Gambar yang digunakan sebagai *input* pada model CNN memiliki ukuran 150x150x3. Angka tiga mengacu pada citra yang memiliki tiga saluran warna, yaitu Merah, Hijau, dan Biru (RGB). Gambar masukan kemudian diproses pada tahap pembelajaran fitur melalui proses konvolusi dan proses. Rancangan ini mencakup dua lapisan konvolusional, masing-masing dengan jumlah filter dan ukuran kernel yang berbeda. Selanjutnya dilakukan proses *flatten* atau transformasi hasil *feature map* dari lapisan *pooling* menjadi vektor. Ini biasanya disebut sebagai tahap *fully connected layer*.

Tabel 3.1 Flowchart Model CNN



Berdasakan tabel 3.1 di atas dijelaskan bahwa arsitektur CNN memiliki dua tahap yaitu *Feature Learning* dan *classification. Feature Learning* adalah teknik yang memungkinkan suatu sistem secara otomatis mengekstraksi representasi gambar ke dalam bentuk angka yang merepresentasikan gambar tersebut. Tahap *classification* adalah tahap dimana hasil *Feature Learning* digunakan untuk klasifikasi berdasarkan *subclass* yang telah ditentukan. Pada konvolusi pertama, terdapat 32 filter dan kernel berukuran 3x3. Kemudian, dilakukan proses *pooling* dengan ukuran 2x2 dan pergeseran *mask* sebanyak dua langkah. Pada tahap

konvolusi kedua, terdapat 64 filter dan kernel berukuran 2x2. Kemudian dilanjutkan ke tahap *flatten*, dimana keluaran dari proses konvolusi yang berupa matriks diubah menjadi vektor dan kemudian digunakan dalam tahap klasifikasi menggunakan MLP (*Multi Layer Perceptrons*) dalam jumlah neuron pada lapisan tersembunyi yang telah ditentukan. Kelas citra diklasifikasikan berdasarkan nilai neuron pada lapisan tersembunyi dengan menggunakan fungsi aktivasi *sigmoid*.

3.5 Rancangan Pengujian

Pengujian ini dilakukan untuk mengevaluasi model yang dihasilkan oleh CNN. Pengujian ini dilakukan dalam dua tahap yaitu tahap: tahap *training* dan tahap *testing*. Selama tahap pelatihan, model CNN diuji dengan menggunakan data latih yang sudah disediakan. Jumlah data latih yang diberikan sebanyak 2300 data gambar, dengan total 1150 gambar per kelas. Data *training* kembali dibagi menjadi dua yaitu *training* dan validasi, yaitu sebanyak 1610 data *training* dan 460 data validasi. Tahap *Testing* merupakan tahap pengujian model yang sudah dilakukan pada tahap pelatihan. Jumlah data *testing* dalam penelitian ini sebanyak 230 data gambar, dengan 115 data gambar per kelas. Pada tahap ini, model diuji dengan gambar yang berbeda untuk menguji apakah model tersebut berfungsi dengan baik dalam mengklasifikasikan gambar.

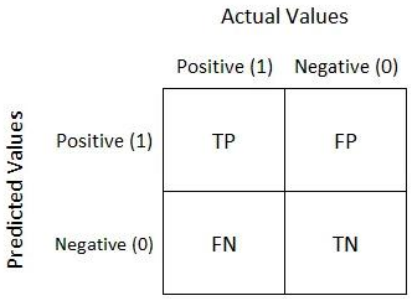
3.6 Evaluasi Model

Langkah selanjutnya adalah evaluasi model *backend Tensorflow* untuk melatih model. Setelah dibersihkan kumpulan data diterjemahkan ke dalam File .h5 (nilai yang dipisahkan koma) dan dimuat ke dalam model pelatihan. File h5 tersebut kemudian diubah menjadi model pembelajaran pasif. Data diproses dengan *Python* yang pada dasarnya menjalankan semua perpustakaan untuk mengompilasi CNN ini. Dengan setiap *literasi* model belajar memprediksi kerusakan pada kendaraan.

3.7 *Confusion matrix*

Kinerja yang baik dari suatu model klasifikasi dapat ditentukan oleh parameter pengukuran kinerjanya, yaitu tingkat akurasi, *recall*, dan presisi. Menghitung faktor-faktor tersebut memerlukan sebuah matriks, yang biasa disebut

sebagai matriks kebingungan (*confusion matrix*). Salah satu *confusion matrix* yang umum digunakan untuk pengukuran dapat ditunjukkan pada Gambar 2.13 [15]



Gambar 3.7 *Confusion matrix*

Berdasarkan gambar 3.4 di atas, matriks memiliki beberapa nilai: *true positive* (TP), *true negative* (TN), *false positive* (FP), dan *false negative* (FN). Serta segala kemungkinan kejadian yang mungkin bernilai positif (P) dan seluruh kemungkinan kejadian yang mungkin bernilai negatif (N). Nilai-nilai tersebut dapat digunakan untuk menghitung akurasi menggunakan persamaan (3.2)

$$Akurasi = \frac{TP+TN}{P+N} \tag{3.2}$$

Akurasi digunakan sebagai parameter untuk mengevaluasi seberapa akurat model melakukan klasifikasi. Sementara itu, untuk menghitung tingkat presisi prediksi kejadian dapat menggunakan persamaan (3.3).

$$Presisi\ prediksi = \frac{TP}{TP+FP} \tag{3.3}$$

Presisi menunjukkan seberapa tepat suatu model memprediksi kejadian positif dalam di berbagai kegiatan prediksi. Perhitungan presisi biasanya sangat berguna Ketika mengembangkan model prediksi curah hujan di suatu daerah. Selain presisi dan akurasi, untuk melihat lebih detail kinerja suatu sistem, kita juga dapat memeriksa *recall* atau sensitivitas sistem terhadap suatu kelas. *Recall* dapat dihitung dengan menggunakan persamaan (3.4).

$$Sensitifitas\ prediksi = \frac{TP}{TP+FN} \tag{3.4}$$