Klasifikasi Kecacatan Ban Untuk Mengendalikan Kualitas Produk Menggunakan Model CNN Dengan Arsitektur VGG-16

Classification Of Tire Defect To Control Product Quality Using Cnn Model With VGG-16 Architecture

1st Alif Ludfi As Shidiq
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
alifludfi@telkomuniversity.ac.id

2nd Efri Suhartono
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
esuhartono@telkomuniversity.ac.id

3rd Sofia Saidah
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
sofiasaidahsfi@telkomuniversity.ac.id

Abstrak—Ban merupakan suatu alat yang berfungsi sebagai penggerak kendaraan. Dalam penggunaan ban terdapat beberapa aspek yang perlu diperhatikan seperti durabilitas, performa, dan keamanan pengguna. Dengan demikian, dalam produksi ban harus memastikan kondisi fisik ban yang sudah sesuai dengan standar tanpa adanya kecacatan. Untuk mengatasi permasalahan tersebut, diperlukan suatu sistem menggunakan komputer untuk menggantikan tenaga manusia dalam melakukan inspeksi permukaan ban. Salah satu sistem yang dapat merealisasikan hal tersebut adalah penggunaan *deep learning*. Terdapat beberapa metode *deep learning*, salah satunya adalah *Convolutional Neural Network* (CNN). CNN dapat menirukan sistem pengenalan citra pada visual cortex manusia, sehingga cocok digunakan untuk identifikasi dan klasifikasi data citra. Simulasi klasifikasi kecacatan ban dilakukan menggunakan Google Colaboratory dengan pengambilan sumber *dataset* dari www.kaggle.com. Proses klasifikasi dimulai dengan memasukkan *dataset* yang berisi jenis-jenis kecacatan ban, setelah itu diproses dalam *hidden layer* CNN. Performa terbaik dari penelitian ini yaitu akurasi 87%, *precision* 81%, *recall* 96%, *f-1 score* 88% dengan parameter yang optimal yang didapatkan dalam penelitian ini diantaranya; *Resize 224x224, Optimizer SGD, Epoch 70, batch size 16,* dan *learning rate 0,0001.*

Kata Kunci— kecacatan ban, klasifikasi, *deep learning*, CNN, VGG-16.

Abstract—Tires are a tool that functions as a vehicle propulsion. In the use of tires, there are several aspects that need to be considered such as durability, performance, and user safety. Thus, in tire production, it is necessary to ensure that the physical condition of the tires is in accordance with the standards without any defects. To overcome these problems, we need a system using a computer to replace human labor in conducting tire surface inspections. One system that can realize this is the use of deep learning. There are several deep learning methods, one of which is Convolutional Neural Network (CNN). CNN can mimic the image recognition system in the human visual cortex, making it suitable for identification and classification of image data. Tire defect classification simulation was carried out using Google Colaboratory and dataset sourced www.kaggle.com. The classification process begins by entering a dataset containing the types of tire defects, after which it is processed in the CNN hidden layer. The best performance of this study is 87% accuracy, 81% precision, 96% recall, f-1 score 88% with the optimal parameters obtained in this study including; Resize 224x224, Optimizer SGD, Epoch 70, batch size 16, and learning rate 0.0001.

Keywords— tire defect, classification, deep learning, CNN, VGG-16

I. PENDAHULUAN

Deteksi kecacatan pada suatu produk sangat berpengaruh penting terhadap kemajuan industri. Pada kehidupan sehari-hari manusia masih menggunakan cara manual untuk membedakan antara ban yang cacat dan tidak cacat. Pemeriksaan visual maupun ban sangat berperan penting untuk menjaga kualitas dan keselamatan berkendara.

Salah satu permasalahan utama yang dihadapi oleh perusahaan yang memproduksi ban yaitu terkait pengendalian kualitas produk cacat [1]. Perusahaan mengharapkan produk dengan kualitas tinggi, sehingga memerlukan proses dan waktu yang cukup lama. Namun, produk ban yang dihasilkan dari proses produk tidak seluruhnya menghasilkan kualitas yang baik . Jika terdapat produk yang tidak berkualitas yang diterima konsumen, akan berdampak pada kepuasan dan loyalitas konsumen [1].

Dengan demikian, perusahaan perlu melakukan pengendalian terhadap kualitas produk ban yang baik maupun cacat sebelum di distribusikan ke pasar. Namun, sistem inspeksi yang umumnya dilakukan oleh industri ban dinilai kurang efektif karena masih dilakukan secara manual dengan tenaga manusia. Hal tersebut menimbulkan resiko *human error* dan membutuhkan waktu yang lama.

Riset terkait penanganan permasalahan terkait pengendalian kualitas produk saat ini sudah dilakukan dengan deteksi otomatis menggunakan teknologi komputer [2], salah satunya dengan menggunakan pengenalan pola pada citra dan klasifikasi [3]. Permasalahan tentang kecacatan produk dapat diselesaikan dengan teknik pengenalan pola berdasarkan citra objek [4]. Hal ini berlaku juga pada deteksi kecacatan pada ban kendaraan. Pendeteksian secara otomatis diharapkan mampu menemukan kecacatan pada setiap produk ban yang diproduksi, sehingga mengurangi kemungkinan produk cacat yang diterima oleh konsumen.

Deep learning merupakan salah satu aspek dalam *Artificial Intelligence* (AI) yang memungkinkan pembelajaran secara mandiri yang dilakukan oleh komputer. *Deep learning* mampu melakukan proses ekstraksi fitur dan klasifikasi pada suatu arsitektur yang sama. *Convolutional Neural Network* (CNN) merupakan salah satu metode *deep learning* yang dapat menyelesaikan permasalahan pengenalan pola dan klasifikasi pada objek citra [5]. CNN melakukan pembelajaran dengan menggunakan fitur *learning* pada proses pengenalan pola citra. CNN menirukan sistem pengenalan citra pada visual cortex manusia, sehingga metode CNN umumnya digunakan dalam identifikasi dan klasifikasi data citra [6]. Kelebihan metode *Convolutional Neural Network (CNN)* dengan menggunakan arsitektur VGG-16 tersebut bisa menampilkan arsitektur yang sangat homogen dan hanya melakukan convolutional 3x3 dan pooling 2x2 dari awal hingga akhir, dan VGG-16 ini memiliki arsitektur yang sangat sedikit dibandingkan dengan arsitektur lain.

Beberapa penelitian telah melakukan riset tentang deteksi pada cacat produk menggunakan berbagai macam teknik di antaranya *machine learning* dan *deep learning* [7]. Penelitian [8] melakukan deteksi cacat produk pengelasan menggunakan metode CNN yang menghasilkan akurasi sistem sebesar 98,75%. Selain itu, deteksi cacat produk botol kaca juga dilakukan dengan menggunakan metode CNN dan mendapatkan akurasi sistem sebesar 99% [4]. Deteksi cacat produk juga dilakukan dengan menggunakan CNN yang dikombinasikan dengan *Long-Short-Term Memory* (LSTM) pada objek kain pabrik dan menghasilkan akurasi sistem sebesar 99% [9]. Komite Nasional Keselamatan Transportasi (KNKT) menyatakan bahwa 80 persen kecelakaan tol dikarenakan pecahnya ban kendaraan dan kasus tersebut didapatkan dari kendaraan umum maupun pribadi, tentunya hal ini menimbulkan tanda tanya pada kualitas ban yang digunakan dan ban tersebut diproduksi oleh pabrik mana [10].

Berdasarkan pada penelitian sebelumnya, metode CNN memberikan kinerja sistem yang baik pada pendeteksian kecacatan produk. Pada penelitian tugas ini, CNN dengan arsitektur VGG-16 diusulkan untuk melakukan klasifikasi terhadap kecacatan ban untuk mengendalikan kualitas produk. Penggunaan metode CNN dengan arsitektur VGG-16 diharapkan mampu menghasilkan kinerja sistem yang baik dalam melakukan klasifikasi pada kecacatan ban.

II. KAJIAN TEORI

A. Kecacatan Ban

Kecacatan ban merupakan salah satu aspek yang muncul ketika kegiatan produksi ban dilakukan. Kecacatan ban perlu diidentifikasi untuk menghindari terjadinya komplain dari pihak konsumen. Terdapat berbagai macam jenis kecacatan pada ban hasil produksi, di antaranya:

a. *Crack buster* adalah cacat berupa keretakan di bagian punggung.
b. *Crack sidewall* adalah cacat ban berupa keretakan ban bagian samping ban.
c. *Dirty mould* adalah cacat ban disebabkan terdapat kotoran yang menempel ban.
d. *Blown sidewall* adalah cacat ban berupa gelembung udara di bagian *sidewall*.
e. *Blown play* adalah cacat ban berupa gelembung udara antara lapisan *ply*.
f. *Internal failure* adalah cacat yang disebabkan oleh tekanan dalam gagal.
g. *Blown tread* adalah cacat ban berupa gelembung udara di bagian *tread.*
h. *Under cure tread* adalah cacat ban berupa ban mentah di bagian *tread.*

Hingga saat ini, kegagalan produksi terhadap kecacatan ban yang diproduksi oleh perusahaan masih sering ditemukan. Padahal proses pembuatan ban memerlukan bahan dan biaya produksi yang cukup banyak. Permasalahan ini membuktikan bahwa pendeteksian ban yang dilakukan oleh produsen kurang efisien, apalagi jika produk yang cacat tersebut sudah sampai pada tangan konsumen, tentunya produk tersebut akan membahayakan konsumen dan nama baik produsen yang kredibilitasnya ditanyakan. Gambar 2.1 menunjukkan contoh kecacatan pada ban.

GAMBAR 2. 1
CITRA BAN CACAT

Penyebab dari kecacatan produksi ban adalah, yang pertama dari bahan pembuatnya sendiri yang kualitasnya buruk, lalu *tread rubber* atau cetakan karet yang tidak sempurna, faktor mesin yang digunakan selama proses produksi, lalu yang terakhir adanya kesalahan dalam proses deteksi untuk *finishing*. Komite Nasional Keselamatan Transportasi (KNKT) menyatakan bahwa 80 persen kecelakaan tol dikarenakan pecahnya ban kendaraan dan kasus tersebut didapatkan dari kendaraan umum maupun

pribadi, tentunya hal ini menimbulkan tanda tanya pada kualitas ban yang digunakan dan ban tersebut diproduksi oleh pabrik mana [10].

B. Pengolahan Citra Digital

Citra merupakan salah satu komponen penting sebuah informasi yang direpresentasikan dalam bentuk visual [15]. Citra merepresentasikan kemiripan dari suatu objek. Terdapat dua jenis citra, yaitu citra analog dan citra digital. Citra analog merupakan citra yang bersifat kontinu yang memiliki rentang nilai intensitas cahaya 0 sampai dengan ~ (tak hingga). Citra digital merupakan citra yang terbentuk dari nilai diskrit dan nilai intensitas cahayanya tergantung pada kedalaman bit yang menyusunnya [13].

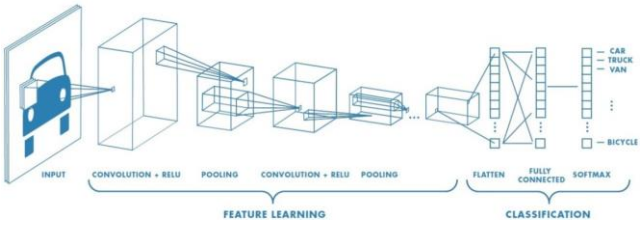
Dalam pengolahan citra digital, terdapat beberapa jenis citra yang dapat diolah menggunakan komputer. Contohnya, citra warna (RGB), citra keabuan (grayscale), dan citra biner. Citra warna (RGB) merupakan citra yang rentang intensitas cahayanya 0 hingga 255. Citra keabuan (grayscale) merupakan citra yang rentang intensitas cahayanya 0 hingga 255. Sedangkan, citra biner merupakan citra yang rentang intensitas cahayanya 0 dan 1 [13].

Pengolahan citra digital merupakan pemrosesan citra menggunakan komputer yang digunakan untuk memanipulasi citra. Beberapa aplikasi yang dilakukan dengan menggunakan pengolahan citra digital meliputi pengenalan pola, *computer vision*, dll.

C. Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) merupakan salah satu jenis dari jaringan syaraf tiruan yang bekerja secara mendalam melalui pembelajaran secara berulang-ulang. CNN pada umumnya digunakan untuk melakukan pengenalan pola pada objek citra. Algoritma CNN melakukan tahapan ekstraksi ciri dan klasifikasi dalam satu proses [12]. CNN memiliki dua lapisan utama, yaitu *feature learning layer* dan *classification layer* [5]. *Feature learning layer* merupakan lapisan yang berperan untuk melakukan ekstraksi ciri pada citra. *Classification layer* merupakan lapisan yang berperan untuk melakukan klasifikasi atau pengelompokan citra berdasarkan label yang telah ditentukan. *Classification layer* akan memproses klasifikasi berdasarkan ciri yang dihasilkan dari lapisan *feature learning*.

Gambar 2.2 menunjukkan arsitektur dari CNN secara umum.



GAMBAR 2. 2
ARSITEKTUR CNN [17]

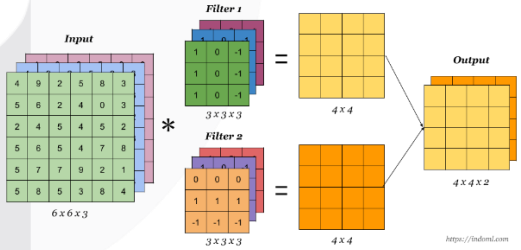
Berdasarkan Gambar 2.2, proses CNN memiliki beberapa lapis tahapan, di antaranya:

1. *Input*: masukan awal berupa citra
2. Lapisan Konvolusi: lapisan yang menghasilkan nilai dari perhitungan antara citra dengan *filter*.
3. Fungsi Aktivasi ReLU: digunakan untuk normalisasi dengan cara mengubah nilai piksel negatif menjadi 0 pada hasil konvolusi.
4. Lapisan *Pooling*: lapisan yang mereduksi ukuran spasial dan jumlah parameter dalam jaringan sehingga mempercepat proses komputasi.
5. Lapisan *Fully connected*: lapisan yang mengintegrasikan fitur dari lapisan sebelumnya.
6. Lapisan *Output*: berupa lapisan yang akan menunjukkan neuron sebanyak jumlah wajah atau kelas yang akan diprediksi.

1. Convolution Layer

Convolution Layer atau lapisan konvolusi merupakan lapisan pertama yang menerima input citra [5]. Pada lapisan ini, dilakukan pengurangan dimensi citra dengan cara menggunakan *filter* untuk melakukan ekstraksi ciri dari citra. Konvolusi dilakukan dengan melakukan perkalian antara piksel dari citra dengan ukuran *filter*. Keluaran atau *output* dari proses konvolusi yaitu berupa *feature map* yang akan digunakan sebagai *input* pada lapisan berikutnya. Lapisan konvolusi dipengaruhi oleh tiga parameter, di antaranya:

a. *Filter*: yaitu suatu matriks berukuran lebih kecil dari citra yang berisi nilai acak sebagai inisialisasi pertama pada arsitektur model.
b. *Stride*: yaitu suatu nilai untuk menentukan jumlah pergeseran *filter* pada proses konvolusi.
c. *Padding*: yaitu suatu nilai yang digunakan untuk mengatasi ukuran spasial dari volume citra yang bekerja dengan cara menambahkan nilai nol di sekeliling matriks citra. Gambar 2.3 menunjukkan operasi konvolusi pada CNN.



GAMBAR 2. 3
VISUALISASI OPERASI KONVOLUSI [18]

Berdasarkan Gambar 2.3, operasi konvolusi dilakukan dengan dilakukan dengan mengalikan antara citra ukuran asli dengan *filter*. Citra berukuran 4x4 dikalikan dengan *filter* berukuran 2x2 dan menghasilkan *output* berukuran 3x3. Dengan demikian, lapisan konvolusi bekerja dengan cara menghitung nilai-nilai unik dan menghasilkan *output* berupa ukuran matriks dengan ukuran lebih kecil.

2. Rectified Linear Unit (ReLU)

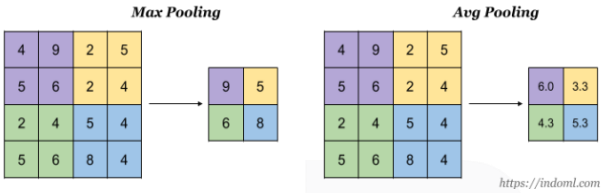
Rectified Linear Unit (ReLU) merupakan suatu fungsi aktivasi pada nilai ambang batas atau *threshold* 0 [6]. ReLU bekerja dengan mengubah nilai matriks yang negatif menjadi nilai 0 (nol). Persamaan (2.1) menunjukkan aktivasi ReLU.

$$f(x) = \max(0, x) \quad (2.1)$$

Berdasarkan persamaan 2.1, *f(x)* akan bernilai 0 jika *x* bernilai negatif, sebaliknya *f(x)* akan mengembalikan nilai *x* jika nilai *x* adalah positif.

3. Pooling

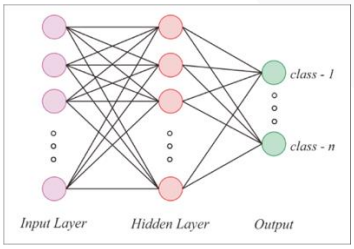
Pooling merupakan lapisan yang berfungsi untuk mereduksi ukuran spasial matriks dan jumlah parameter guna mempercepat proses komputasi [5]. *Pooling* dilakukan dengan cara mengambil sebagian nilai dari *feature map*. *Pooling* memiliki dua jenis cara, yaitu *max-pooling* dan *average-pooling*. *Max-pooling* bekerja dengan cara mengambil nilai paling besar pada area *stride* yang telah ditentukan. Sedangkan *average-pooling* bekerja dengan cara menghitung rata-rata nilai yang ada pada area *stride*. Gambar 2.4 menunjukkan ilustrasi dari lapisan *pooling*.



GAMBAR 2. 4
ILUSTRASI *POOLING* [18]

4. Fully Connected Layer

Fully Connected Layer merupakan lapisan yang digunakan untuk memproses klasifikasi berdasarkan nilai-nilai yang telah terkumpul pada tahap *pooling* [5]. Nilai-nilai yang dihasilkan di lapisan *pooling* akan dijadikan menjadi satu barisan (*flatten*), pada setiap nilainya akan menjadi neuron untuk dilatih dan diklasifikasikan. Selanjutnya, neuron yang telah dilatih akan menghasilkan *output* berupa kelas-kelas sebanyak jumlah kelas yang telah ditentukan pada saat *input*. Gambar 2.5 menunjukkan ilustrasi dari *fully connected layer*.

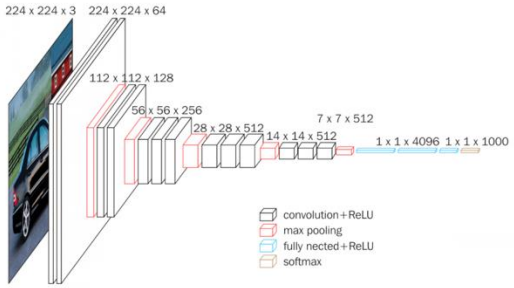


GAMBAR 2. 5
ILUSTRASI *FULLY CONNECTED LAYER* [19]

Berdasarkan Gambar 2.5, *fully connected layer* pada dasarnya bekerja dengan cara melakukan pelatihan menggunakan jaringan syaraf tiruan berganda.

D. Arsitektur VGG-16

VGG-16 merupakan salah satu jenis dari arsitektur CNN yang memiliki hidden layer berjumlah 16 buah [16]. Pada VGG-16, proses konvolusi dilakukan dengan menggunakan *filter* dengan jumlah 3x3, *stride* sejumlah 1 dan *max pooling* sejumlah 2x2. *Input* citra pada VGG-16 berukuran 224x224x3 dan akan melalui reduksi pada blok awal yang menghasilkan 224x224x64 hingga blok akhir sejumlah 7x7x512. Gambar 2.6 menunjukkan arsitektur dari VGG-16. Keunggulan model ini menampilkan arsitektur yang sangat homogen yang hanya melakukan convolutional 3x3 dan pooling 2x2 dari awal hingga akhir.



GAMBAR 2. 6
ARSITEKTUR VGG-16 [20]

E. Confusion Matrix

Confusion matrix merupakan suatu alat untuk mengukur kinerja dari model klasifikasi yang telah dibangun [14]. *Confusion matrix* menjadi salah satu evaluator untuk mengukur seberapa baik model yang dibangun mampu mengenali kelas-kelas yang berbeda. Terdapat 4 parameter pada *confusion matrix*, di antaranya *True Positif* (TP), *False Positif* (FP), *True Negatif* (TN), dan *False Negatif* (FN). TP merupakan jumlah kelas yang terklasifikasi benar sebagai kelas benar oleh sistem klasifikasi. FP merupakan jumlah kelas yang terklasifikasi salah sebagai kelas benar oleh sistem klasifikasi. FN merupakan jumlah kelas benar yang terklasifikasi salah sebagai kelas salah oleh sistem klasifikasi. TN merupakan jumlah kelas salah yang terklasifikasi benar sebagai kelas salah oleh sistem klasifikasi. Tabel 2.1 menunjukkan gambaran dari *confusion matrix*.

TABEL 2. 1
CONFUSION MATRIX

Kelas Sebenarnya	**Kelas Sebenarnya**	
	Good	*Defect*
Good	TP	FN
Defect	FP	TN

Berdasarkan hasil *confusion matrix*, maka dapat dihitung beberapa indikator evaluasi seperti nilai akurasi, presisi, *recall* dan *f1-score*. Akurasi merupakan tingkat ketepatan prediksi benar dari keseluruhan data. Presisi merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan hasil yang diprediksi positif. *Recall* merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan data yang benar positif. *F1-score* merupakan kalkulasi evaluasi dalam pencarian informasi yang menggabungkan *recall* dan presisi. Perhitungan akurasi, presisi, *recall* dan *F1-score* ditunjukkan pada persamaan (3.1), (3.3), (3.4), dan (3.5) [11].

$$\text{Akurasi} = \frac{TP + TN}{TP + TN + FP + FN} \quad (3.1)$$

$$\text{Presisi} = \frac{TP}{TP + FP} \quad (3.2)$$

$$Recall = \frac{TP}{TP + FN} \quad (3.3)$$

$$F1 - score = 2\,x\,\frac{(Recall\ x\ Presisi)}{(Recall + Presisi)} \quad (3.4)$$

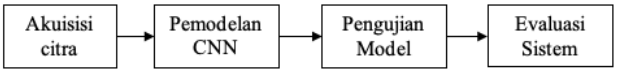
Keterangan:

- TP adalah True Positive, yaitu jumlah data positif yang terklasifikasi dengan benar oleh sistem.
- FP adalah False Positive, yaitu jumlah data positif namun terklasifikasi salah oleh sistem.
- FN adalah False Negative, yaitu jumlah data negatif namun terklasifikasi salah oleh sistem.
- TN adalah True Negative, yaitu jumlah data negatif yang terklasifikasi dengan benar oleh sistem.

III. METODE

A. Desain Sistem

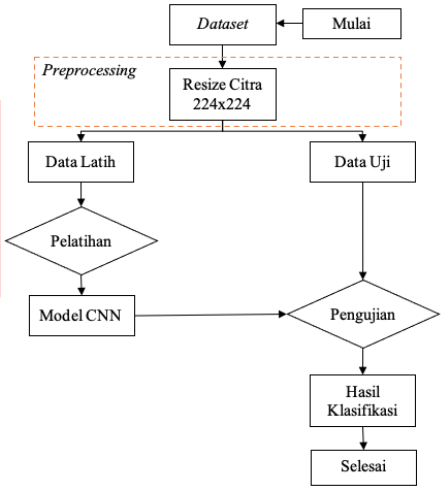
Pada penelitian tugas akhir ini, deteksi kecacatan ban akan dilakukan dengan menggunakan metode CNN dengan arsitektur VGG-16. Penelitian dimulai dengan melakukan akuisisi data yaitu berupa citra ban normal dan cacat. Selanjutnya, sekumpulan data citra ban (*dataset*) akan dimodelkan menggunakan CNN dengan arsitektur VGG-16. Selanjutnya pengujian terhadap model klasifikasi dilakukan untuk mengetahui hasil kinerja sistem. Klasifikasi pada penelitian ini menggunakan dua kelas, yaitu kelas normal dan kelas cacat. Evaluasi sistem dapat dilakukan berdasarkan hasil pengujian sistem, dengan demikian kinerja sistem dapat ditentukan baik maupun buruknya. Pada penelitian ini, pengujian sistem dilakukan dengan menggunakan parameter *resize*, *optimizer*, *batch size*, *epoch* dan *initial learning rate* yang berbeda-beda untuk mengetahui seberapa jauh sistem mampu bekerja optimal dan maksimal. Gambar 3.1 menunjukkan desain sistem yang dirancang pada penelitian ini.



GAMBAR 3. 1
DESAIN SISTEM

B. Diagram Alir Sistem

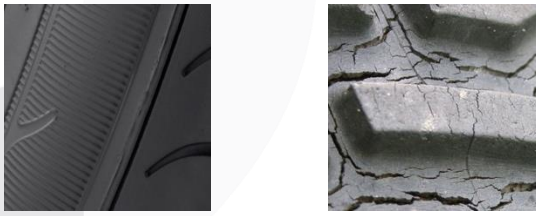
Deteksi kecacatan ban yang dilakukan dengan menggunakan metode CNN dengan arsitektur VGG-16 diproses menggunakan beberapa tahap. Pertama, *dataset* citra berwarna akan digunakan sebagai *input* sistem. Selanjutnya, seluruh *dataset* akan di *resize* menjadi ukuran 224x224 piksel. Hal ini dilakukan karena *input* pada proses CNN dengan arsitektur VGG-16 berupa citra berukuran 224x224x3. Selanjutnya, akan dilakukan pembagian data menjadi data latih dan data uji. Data latih akan menjadi *input* dari pelatihan CNN dengan arsitektur VGG-16. Selanjutnya, setelah model dibangun, data uji digunakan sebagai data untuk melakukan uji kinerja sistem pada model CNN yang telah dibangun. Selanjutnya, berdasarkan hasil uji, hasil klasifikasi dapat didapatkan melalui perhitungan akurasi, presisi, *recall* dan *f1-score*. Gambar 3.2 menunjukkan diagram alir sistem pada penelitian ini.



GAMBAR 3. 2
DIAGRAM ALIR SISTEM

1. Dataset

Dataset pada penelitian ini menggunakan citra ban yang telah dilabeli kelas normal dan kelas cacat. *Dataset* yang digunakan bersumber dari www.kaggle.com dengan jumlah data sebanyak 1039 citra ban. Dataset terbagi menjadi dua bagian yaitu data latih sebanyak 831 citra dan data uji sebanyak 208 citra. Gambar 3.3 (a) dan (b) menunjukkan contoh citra ban kelas normal dan kelas cacat.



(a) (b)

GAMBAR 3. 3
CONTOH CITRA BAN, (A) KELAS NORMAL, (B) KELAS CACAT

2. Preprocessing

Preprocessing dilakukan untuk mengatur kebutuhan citra yang diperlukan sebelum masuk ke dalam proses pemodelan klasifikasi. Pada penelitian ini, ukuran seluruh *dataset* citra akan di *resize* menjadi ukuran 224x224 piksel. Hal ini dilakukan untuk menyesuaikan input dari arsitektur VGG-16, *input* awal berupa citra berwarna berukuran 224x224 piksel. Selanjutnya, citra yang telah di *resize* akan menjadi *input* di proses pelatihan maupun pengujian model klasifikasi yang akan dibangun.

3. Pelatihan CNN Arsitektur VGG-16

Tahapan selanjutnya yaitu melakukan pelatihan pada arsitektur VGG-16 berdasarkan data latih citra ban yang telah di *resize*. Arsitektur VGG-16 memiliki beberapa tahap yang melibatkan banyak lapisan, meliputi lapisan konvolusi, lapisan ReLU, lapisan *pooling*, serta lapisan *fully connected*. VGG-16 memiliki 13 proses konvolusi, 3 proses *fully connected*, dan 1 proses *softmax* di akhir proses. Tabel 3.1 menunjukkan detail arsitektur VGG-16 yang digunakan pada penelitian ini.

TABEL 3. 1
ARSITEKTUR VGG-16

No	Lapisan	Detail
1	*Input*	224×224×3
2	*Convolution*	64 3×3×3 konvolusi, *stride* [1 1], *padding* [1 1 1 1]
3	ReLU	ReLU
4	*Convolution*	64 3×3×64 konvolusi, *stride* [1 1], *padding* [1 1 1 1]
5	ReLU	ReLU
6	*Max Pooling*	2×2 *max pooling*, *stride* [2 2], *padding* [0 0 0 0]
7	*Convolution*	128 3×3×64 konvolusi, *stride* [1 1], *padding* [1 1 1 1]
8	ReLU	ReLU
9	*Convolution*	128 3×3×128 konvolusi, *stride* [1 1], *padding* [1 1 1 1]
10	ReLU	ReLU
11	*Max Pooling*	2×2 *max pooling*, *stride* [2 2], *padding* [0 0 0 0]
12	*Convolution*	256 3×3×128 konvolusi, *stride* [1 1], *padding* [1 1 1 1]
13	ReLU	ReLU
14	*Convolution*	256 3×3×256 konvolusi, *stride* [1 1], *padding* [1 1 1 1]
15	ReLU	ReLU
16	*Convolution*	256 3×3×256 konvolusi, *stride* [1 1], *padding* [1 1 1 1]
17	ReLU	ReLU
18	*Max Pooling*	2×2 *max pooling*, *stride* [2 2 3.1.1 *ding* [0 0 0 0]
19	*Convolution*	512 3×3×256 konvolusi, *stride* [1 1], *padding* [1 1 1 1]
20	ReLU	ReLU
21	*Convolution*	512 3×3×512 konvolusi, *stride* [1 1], *padding* [1 1 1 1]
22	ReLU	ReLU
23	*Convolution*	512 3×3×512 konvolusi, *stride* [1 1], *padding* [1 1 1 1]
24	ReLU	ReLU
25	*Max Pooling*	2×2 *max pooling*, *stride* [2 2], *padding* [0 0 0 0]
26	*Convolution*	512 3×3×512 konvolusi, *stride* [1 1], *padding* [1 1 1 1]
27	ReLU	ReLU
28	*Convolution*	512 3×3×512 konvolusi, *stride* [1 1], *padding* [1 1 1 1]
29	ReLU	ReLU
30	*Convolution*	512 3×3×512 konvolusi, *stride* [1 1], *padding* [1 1 1 1]
31	ReLU	ReLU
32	*Max Pooling*	2×2 *max pooling*, *stride* [2 2], *padding* [0 0 0 0]
33	*Fully Connected*	4096 *fully connected layer*
34	ReLU	ReLU
35	*Dropout*	50% dropout
36	*Fully*	4096 *fully connected layer*
40	*Softmax*	*Softmax*
41	*Classification Output*	2 *output*

4. Pengujian Sistem

Tahapan pengujian dilakukan untuk mengukur kinerja dari model klasifikasi yang telah dibangun. Data uji akan menjadi *input* dalam proses pengujian untuk diklasifikasikan melalui model pelatihan yang telah dibangun. Selanjutnya, berdasarkan hasil pengujian sistem dapat diperhitungkan evaluasi kinerja dari model yang telah dibangun.

5. Evaluasi Kinerja Sistem

Evaluasi kinerja sistem dilakukan untuk mengukur seberapa baik model yang dibangun. Pada penelitian ini, evaluasi kinerja akan dihitung berdasarkan hasil *confusion matrix* dari proses pengujian. Kinerja sistem diukur melalui 4 indikator, di antaranya akurasi, presisi, *recall* dan *f-1 score*

a. **Akurasi**: Indikator pertama adalah Akurasi. Akurasi dilakukan untuk menghitung rasio prediksi benar (positif dan negatif) terhadap keseluruhan [11]. Nilai akurasi dapat diperoleh dengan rumus berikut.

$$\text{Akurasi} = \frac{TP + TN}{TP + TN + FP + FN}$$

TP (True Positive) adalah jumlah data positif yang terdeteksi benar, TN (True Negative) adalah jumlah negatif yang terdeteksi benar. FP (False Positive) adalah jumlah data negatif namun terdeteksi sebagai data positif, dan FN (False Negative) adalah jumlah data positif yang terdeteksi sebagai data negatif.

b. **Presisi**: Indikator kedua adalah Presisi. Presisi adalah jumlah perkiraan data yang benar dibandingkan dengan keseluruhan hasil yang diperkirakan positif. [11]. Nilai presisi dapat dirumuskan sebagai berikut.

$$\text{Presisi} = \frac{TP}{TP + FP}$$

TP (True Positive) adalah jumlah data positif yang terdeteksi benar, dan FP (False Positive) adalah jumlah data positif yang terdeteksi salah.

c. ***Recall***: Indikator ketiga adalah *Recall*. *Recall* adalah rasio prediksi benar positif terhadap keseluruhan data yang benar positif [11]. Nilai *recall* dapat dirumuskan sebagai berikut.

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

TP (True Positive) adalah jumlah data positif yang terdeteksi benar dan FN (False Negative) adalah jumlah diprediksi negatif yang terdeteksi benar.

d. ***F1-score***: Indikator keempat adalah *F1-score*. *F1-score* adalah kalkulasi evaluasi dalam pencarian informasi yang menggabungkan *recall* dan presisi. Pada situasi tertentu, nilai *recall* dan presisi dapat memiliki bobot yang berbeda [11]. Nilai *F1-score* dapat dirumuskan sebagai berikut.

$$F1 - score = 2\,x\,\frac{(Recall\ x\ Presisi)}{(Recall + Presisi)}$$

IV. HASIL DAN PEMBAHASAN

A. Spesifikasi Perangkat

Spesifikasi Perangkat Keras pada penelitian ini menjelaskan tentang spesifikasi perangkat yang digunakan untuk menguji sistem yang telah dibangun. Tabel 4.1 menunjukkan spesifikasi perangkat keras yang digunakan pada penelitian ini.

Tabel 4.1 Spesifikasi Perangkat Keras

No.	Kebutuhan	Jenis
1	Sistem Operasi	Macbook Pro macOS Catalina Version 10.15.7
2	Processor	2,3 GHz Quad-Core Intel Core i7
3	Graphics	Intel HD Graphics 4000 1536 MB
4	RAM	8 GB
5	Software	Google Colaboratory Microsoft Word dan Excel 365

B. Pengujian Sistem

Model sistem yang telah dirancang sebelumnya, dilakukan pengujian dalam Tugas Akhir ini menggunakan Google Colaboratory. *Dataset* dengan jumlah data 1028 citra, dan dataset terbagi menjadi dua bagian yaitu data latih sebanyak 703 citra dan data uji sebanyak 315 citra. Maka dari itu ada dua proses utama dalam pengujian Tugas Akhir ini yaitu *training* data dan *testing* data. Hasil performa pada pengujian ini adalah data akurasi, presisi, *recall*, *f1- score*, dan grafik hasil pengujian.

C. Skenario Pengujian Sistem

Pada tahap perancangan sistem dalam Tugas Akhir ini, telah dilakukan lima skenario pengujian. Skenario pertamanya adalah membandingkan pengaruh ukuran citra yang di resize pada algoritma sistem yang telah dibangun untuk mendapatkan nilai akurasi terbaik. Pada skenario kedua, dilakukan perbandingan terhadap optimizer terbaik berdasarkan ukuran citra terbaik yang didapatkan pada skenario sebelumnya. Pada skenario pengujian ketiga, dilakukan perbandingan terhadap epoch untuk menentukan berapa kali model neural network telah melihat dataset secara keseluruhan. Pada skenario keempat, dilanjutkan dengan pengujian nilai *batch size* yang mempengaruhi hasil performa sistem. Pengujian skenario yang kelima adalah membandingkan nilai *learning rate* yang dapat mengoptimalkan sistem. Beberapa nilai *learning rate* yang berbeda diuji coba untuk mendapatkan nilai performa yang terbaik. Maka dengan mendapatkan nilai *resize* citra, *optimizer*, *learning rate*, *epoch*, dan *batch size* terbaik sistem yang paling optimal pun didapatkan dalam Tugas Akhir ini.

D. Hasil Pengujian Sistem

Hasil pengujian sistem dilakukan dengan mengubah nilai parameter dari setiap skenario yang diujikan. Parameter dalam pengujian ini adalah *resize* citra, *optimizer*, *epoch*,*batch size*, *learning rate* dengan pengujian nilai sebagai berikut. Pada skenario *resize*, citra dari *dataset* disamakan ukurannya menjadi 64×64, 224×224 dan 256×256. Lalu pada skenario *optimizer* dilakukan dengan mengubah *optimizer* pada sistem, ada 4 *optimizer* yang diuji coba yaitu Adam, Nadam, RMSProp, dan SGD. Di tahap pengujian *epoch* nilai *epoch* diubah 11 kali menjadi 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, dan 150. Selanjutnya, diujikan nilai *batch size* dengan nilai 8, 16, dan 32. Dan terakhir, dilakukan pengujian skenario *learning rate* diuji coba dengan beberapa nilai yaitu 0,1; 0,01; 0,001; dan 0,0001. Dengan pengubahan nilai parameter tersebut disimpulkan nilai parameter terbaik untuk sistem CNN dengan arsitektur dasar yang telah dimodifikasi agar mendapatkan nilai performa yang optimal pada penelitian klasifikasi kecacatan pada ban.

1. Pengujian Resize

Skenario pengujian pertama dalam penelitian ini adalah mencari nilai citra terbaik yang akan diolah pada sistem. Maka dilakukan proses *pre-processing* yaitu melakukan *resize* pada citra dengan pengubahan ukuran citra sebagai berikut, 64×64, 224×224 dan 256×256. Dengan *hyper* parameter awal digunakan *optimizer* RMSProp, , *epoch* 75, *batch size* 16, dan *learning rate* 0,0001. Hasil pengujian *resize* dapat dilihat pada Tabel 4.2.

TABEL 4.2
HASIL PENGUJIAN *RESIZE*

Resize	Akurasi	Presisi	*Recall*	*F1-score*
64×64	76%	70%	86%	77%
224×224	87%	85%	92%	88%
256×256	72%	79%	57%	67%

Berdasarkan tabel hasil pengujian di atas dan dari ketiga percobaan di atas, grafik perbandingan akurasi pada saat *testing* data dan *training* data yang mendekati stabil diperoleh saat *resize* citra 224x224. Nilai akurasi dari *resize* citra 224x224 pun cukup baik yaitu dengan akurasi 87%. Sehingga untuk pengujian selanjutnya nilai citra ini dijadikan *fixed* parameter.

2. Pengujian Optimizer

Skenario kedua pada pengujian sistem ini adalah dengan mengubah *optimizer* sistem untuk mendapatkan *optimizer* yang paling optimal pada sistem. Setelah pengujian skenario pertama, digunakan *fixed* parameter citra *resize* terbaik yaitu 224x224. Dan parameter yang digunakan pada skenario pengujian ini yaitu *learning rate* 0,0001; *epoch* 75, dan *batch* size 16. Hasil pengujian *optimizer* dapat dilihat pada Tabel 4.3.

TABEL 4.3
HASIL PENGUJIAN *OPTIMIZER*

Optimizer	Akurasi	Presisi	*Recall*	*F1-score*
Adam	88%	88%	87%	87%
Nadam	88%	90%	88%	89%
RMSProp	87%	85%	92%	88%
SGD	77%	75%	73%	74%

Berdasarkan Tabel 4.3 di atas, didapatkan *optimizer* dengan akurasi terbaik pada sistem adalah dengan *optimizer* Adam yaitu dengan akurasi 88%. Namun grafik hasil pengujian dari *optimizer* berikut menampilkan hasil yang kurang stabil Nilai akurasi dari data uji dan data latih menampilkan terjadinya *over fitting*. Kondisi ini terjadi saat sistem bekerja optimal pada *testing* data, namun hasil yang ditunjukkan sistem saat *training* data cukup berbeda, sehingga ada gap hasil akurasi. Dari keempat percobaan di atas, *optimizer* terbaik menurut nilai akurasi, presisi, *recall*, dan *f1-score* adalah SGD. Grafik perbandingan pengujian data latih dan ujipun menunjukkan *optimizer* SGD adalah yang paling stabil dalam pengujian ini. Sehingga perfomansi yang dihasilkan oleh *optimizer* ini lebih baik, dengan nilai akurasi 77%. Dengan memanfaatkan estimasi momen pertama dan kedua dari *epoch*

membuat SGD menjadi efisien secara komputasi. Sehingga untuk pengujian skenario selanjutnya *optimizer SGD* digunakan sebagai *fixed* parameter.

3. Pengujian Epoch

Pengujian ketiga dalam skenario adalah pengujian nilai *epoch*. Pada pengujian ini ada beberapa nilai *epoch* yang diujikan yaitu 50, 60, 70, 80, 90, 100,

110, 120, 130, 140, 150. Banyaknya nilai *epoch* yang diujikan terjadi karena tidak ada ketentuan secara pasti nilai yang paling optimal. Setiap citra memiliki karakteristiknya masing-masing yang membuat nilai *epoch* yang cocok pada *dataset* citra pun berbeda. Dan di skenario pengujian ini *fixed* parameter yang digunakan adalah *resize* citra berukuran 224x224, *optimizer* SGD, dan *learning rate* 0,0001; *Hyper* parameter yang digunakan adalah *batch size* 16. Hasil pengujian *epoch* dapat dilihat pada Tabel 4.4.

TABEL 4.4
HASIL PENGUJIAN *EPOCH*

Epoch	Akurasi	Presisi	*Recall*	*F1-score*
50	76%	70%	91%	79%
60	82%	75%	94%	83%
70	87%	81%	96%	88%
80	91%	89%	94%	91%
90	79%	84%	73%	78%
100	87%	88%	88%	88%
110	90%	92%	89%	91%
120	88%	88%	90%	89%
130	85%	87%	84%	86%
140	80%	79%	84%	82%
150	89%	86%	94%	90%

Berdasarkan hasil performa dari pengujian *epoch* pada tabel 4.4, pengujian*epoch* terbaik diperoleh saat *epoch* bernilai 70. Hasilgrafik pengujian yang diperoleh juga cukup optimal, karena nilai akurasi saat *testing* data dan *training* data cukup berhimpitan yang menandakan sistem sudah optimal dalam melakukan pengujiannya. Karena nilai *epoch* merupakan jumlah putaran berapa kali sistem melakukan *training* pada *dataset* yang telah dikelompokkan ke dalam *batch*. Semakin besar nilai *epoch* maka semakin banyak sistem mempelajari *dataset* yang ada. Maka dari itu nilai *epoch* terbaik adalah 70, dan nilai ini dijadikan *fixed* parameter sistem yang ada.

4. Pengujian Batch Size

Skenario keempat dalam pengujian sistem ini adalah pengujian batch size. Pada pengujian ini, ada 3 nilai batch size yang dicoba yaitu 8, 16, dan 32. Fixed parameter dalam pengujian ini menggunakan resize citra berukuran 224x224, optimizer SGD, learning rate 0,0001; dan epoch 70. Hasil pengujian batch size dapat dilihat pada Tabel 4.5.

TABEL 4.5
HASIL PENGUJIAN *BATCH SIZE*

Batch Size	Akurasi	Presisi	*Recall*	*F1-score*
8	78%	85%	75%	80%
16	87%	81%	96%	88%
32	80%	85%	80%	82%

Berdasarkan tabel di atas dapat dilihat bahwa nilai *batch* size relatif optimal dengan *fixed* parameter yang telah digunakan. Sistem sudah stabil dan memiliki nilai akurasi yang sangat optimal. *Batch size* adalah jumlah citra dalam *dataset* yang dikelompokkan untuk diuji pada sistem. Setiap satu kelompok masuk secara bersamaan lalu dilanjutkan kelompok selanjutnya. Maka nilai optimalnya pada setiap sistem tidaklah pasti. Berdasarkan pengujian yang telah dilakukan, nilai performa terbaik untuk diterapkan pada sistem Tugas Akhir adalah dengan menggunakan *batch size* 16. Untuk itu *fixed* parameter terakhir pada sistem ini adalah nilai *batch size* 16 dengan nilai akurasi 87%.

5. Pengujian Learning Rate

Skenario terakhir dalam pengujian sistem ini adalah pengujian *learning rate kemudian* dipilih nilai *learning rate* yang paling stabil pada sistem yang dibangun. Dari grafik tersebut terdapat sumbu horizontal yang merupakan Epoch dan sumbu vertikal adalah hasilnya. *Learning rate* yang stabil ditandai dengan grafik yang tidak overfitting maupun underfitting. Maka ada beberapa pengujian nilai yaitu 0,1; 0,01; 0,001; dan 0,0001. Nilai performa dan grafik hasil pengujian mempengaruhi nilai *learning rate* yang dipilih. Dalam pengujian ini digunakan *fixed* parameter *resize* citra 224×224 dan *optimizer* SGD. Juga *hyper* parameter pengujian lainnya seperti *epoch* 70 dan *batch size* 16. Hasil pengujian *learning rate* dapat dilihat pada Tabel 4.6.

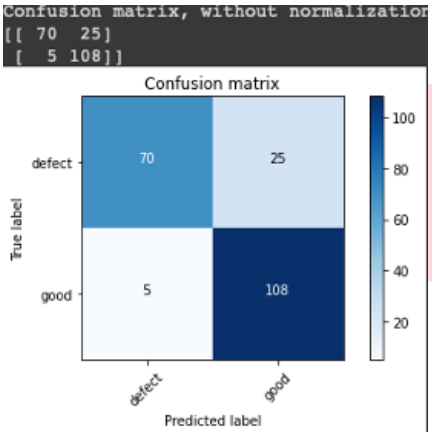
TABEL 4.6
HASIL PENGUJIAN *LEARNING RATE*

Learning Rate	Akurasi	Presisi	*Recall*	*F1-score*
0,1	67%	83%	32%	46%
0,01	84%	96%	72%	82%
0,001	79%	91%	66%	76%
0,0001	87%	81%	96%	88%

Berdasarkan tabel di atas, akurasi nilai *learning rate* terbaik untuk sistem adalah saat nilainya 0,0001. Grafik akurasi pada *learning rate* 0,0001 tersebut cukup stabil. Nilai *learning rate* adalah bobot waktu untuk sistem dalam melakukan koreksi pada proses *training* data. Untuk itu semakin kecil nilai *learning rate* maka semakin cepat sistem melakukan proses pengoreksian. Maka nilai terbaik *learning rate* untuk sistem melakukan proses pengoreksian berdasarkan data akurasi, presisi, *recall*, *f1-score*, dan grafik hasil pengujian yang paling stabil adalah 0,0001.Dengan nilai akurasi 87%, *learning rate* 0,0001 menjadi *fixed* parameter dalam sistem.

E. Hasil Terbaik

Berdasarkan lima skenario pengujian yang telah dilakukan, didapatkan 5 nilaiparameter yang paling optimal untuk sistem yang dirancang. Parameter tersebut adalah ukuran *resize* citra 224×224, jenis *optimizer* SGD, nilai *epoch* 70, nilai *batch size* 16 dan nilai *learning rate* 0,0001;. Dengan menggunakan parameter tersebut, sistem yang dirancang berhasil memperoleh nilai akurasi terbaik sebesar 87%. Berikut adalah model *Confusion Matrix* hasil pengujian sistem dapat dilihat pada Gambar 4.1.



GAMBAR 4.26
CONFUSION MATRIX HASIL PENGUJIAN TERBAIK

Hasil confusion matrix menunjukkan bahwa dari 208 citra yang digunakan untuk validasi data, sebanyak 178 citra terdeteksi benar sesuai kelasnya masing-masing, sedangkan 30 citra lainnya terdeteksi salah.

V. KESIMPULAN

A. Kesimpulan

Dari hasil beberapa skenario uji coba penelitian yang telah dilakukan, dapat diambil kesimpulan, diantaranya:

1. Pengujian memberikan hasil klasifikasi yang cukup baik dibandingkan dengan dataset asli. Hal ini ditunjukkan pada nilai akurasi sebesar 87%.
2. Akurasi tertinggi diperoleh saat sistem menggunakan kombinasi parameter sebagai berikut, resize citra 224×224, optimizer SGD, epoch 70, batch size 16 dan learning rate 0,0001;. Dengan nilai akurasi sebesar 87%. Sistem juga stabil dengan perbandingan pengujian testing data dan training data cukup berhimpitan atau tidak terjadi overfitting dan underfitting.
3. Uji coba dengan jumlah epoch lebih banyak maupun lebih sedikit tidak memberikan pengaruh yang signifikan terhadap akurasi. Namun, jumlah epoch yang lebih banyak memberikan waktu proses yang lebih banyak.

B. Saran

Pengembangan sistem klasifikasi kerusakan ban melalui citra menggunakan preprocessing dan CNN memberikan hasil yang baik. Pada penelitian berikutnya, teknik preprocessing lain dapat digunakan untuk meningkatkan kualitas citra dan mempertajam objek penelitian sebelum di proses menggunakan CNN. Hal perlu dilakukan untuk mendapatkan ciri citra yang lebih jelas dan detail sebelum memasuki proses pelatihan.

REFERENSI

[1] Khanna, A., Kishore, A., & Jaggi, C. K. (2017). Strategic Production Modeling for Defective Items with Imperfect Inspection Process, Rework, and Sales Return Under Two-Level Trade Credit. *International Journal of Industrial Engineering Computations*, 85–118. https://doi.org/10.5267/j.ijiec.2016.7. 001

[2] Carrillo-Gutierrez, T., Martínez, R. M. R., de La Riva Rodríguez, J., & Sanchez-Leal, J. (2016). Relevant aspects of human error and its effect on the quality of the product. Study in the maquiladora industry. *Advances in Intelligent Systems and Computing*, *490*, 475–485. https://doi.org/10.1007/978-3-319-41697-7_42

[3] Rahmayuna, N., Rahardwika, D. S., Sari, C. A., Setiadi, D. R. M., & Rachmawanto, E. H. (2018, November). Pathogenic Bacteria Genus Classification using Support Vector Machine. *2018 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*.

[4] Wang, J., Fu, P., & Gao, R. X. (2019). Machine vision intelligence for product defect inspection based on deep learning and Hough transform. *Journal of Manufacturing Systems*, *51*(December 2018), 52–60. https://doi.org/10.1016/j.jmsy.2019.03.002

[5] Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y., & Alsaadi, F. E. (2017). A survey of deep neural network architectures and their applications. *Neurocomputing*, *234*(October 2016), 11–26. https://doi.org/10.1016/j.neucom.2016.12.038

[6] Agarap, A. F. (2018). *Deep Learning using Rectified Linear Units (ReLU)*. http://arxiv.org/abs/1803.08375

[7] Rahmayuna, N., Adi, K., & Kusumaningrum, R. (2021, December). Tableware Ceramics Defect Detection Using Morphological Operation Approach. *2021 4th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*.

[8] Deng, H., Cheng, Y., Feng, Y., & Xiang, J. (2021). *SS symmetry Industrial Laser Welding Defect Detection and Image Defect*.

[9] Zhao, Y., Hao, K., He, H., Tang, X., & Wei, B. (2020). A visual long-short-term memory based integrated CNN model for fabric defect image classification. *Neurocomputing*, *380*, 259–270. https://doi.org/10.1016/j.neucom.2019.10.067

[10] Hutomo, Di. (2019, July 19). *Jika Kecelakaan Karena Ban Pecah Mengakibatkan Korban Luka dan Motor Rusak.* . https://www.hukumonline.com/klinik/detail/ulasan/lt5d29998826f3b/jika-kecelakaan-karena-ban-pecah-mengakibatkan-korban-luka-dan-motor-rusak

[11] D. Powers, "Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation," 2007.

[12] Bouwmans, T., Javed, S., Sultana, M., & Jung, S. K. (2019). Deep neural network concepts for background subtraction: A systematic review and comparative evaluation. *Neural Networks*, *117*, 8–66. https://doi.org/10.1016/j.neunet.2019.04.024

[13] Andono, P. N., & Sutojo, T. (2015). *Konsep Pengolahan Citra Digital*. Penerbit Andi.

[14] Xu, J., Zhang, Y., & Miao, D. (2020). Three-way confusion matrix for classification: A measure driven view. *Information Sciences*, *507*, 772–794. https://doi.org/10.1016/j.ins.2019.06.064

[15] Padmo, A. A., & Murinto, M. (2016). SEGMENTASI CITRA BATIK BERDASARKAN FITUR TEKSTUR MENGGUNAKAN METODE FILTER GABOR DAN K-MEANS CLUSTERING #1. *Jurnal Informatika*, *10*(1).

[16] Habib, G., & Qureshi, S. (2020). Optimization and acceleration of convolutional neural networks: A survey. *Journal of King Saud University - Computer and Information Sciences*, *xxxx*. https://doi.org/10.1016/j.jksuci.2020.10.004

[17] https://medium.com/@samuelsena/pengenalan-deep-learning-part-7-convolutional-neural-network-cnn-b003b477dc94

[18] https://indoml.com/2018/03/07/student-notes-convolutional-neural-networks-cnn-introduction/

[19] https://www.researchgate.net/figure/Fully-connected-layer_fig3_343263135

[20] https://medium.com/@nutanbhogendrasharma/deep-convolutional-networks-vgg16-for-image-recognition-in-keras-a4beb59f80a7