

***DEEP LEARNING MENGGUNAKAN TENSORFLOW
DAN CONVOLUTIONAL NEURAL NETWORK
UNTUK PENDETEKSIAN KEMASAN
BISKUIT NEXTAR YANG RUSAK***

SKRIPSI



oleh:

Ferry Syarif Fuddin
2120190398

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS NAHDLATUL ULAMA SUNAN GIRI
2023**

PERNYATAAN

Saya menyatakan bahwa skripsi ini bebas plagiat, dan apabila saya di kemudian hari terbukti terdapat plagiat dalam skripsi ini, maka saya bersedia menerima sanksi sesuai ketentuan peraturan perundang-undangan.

Bojonegoro, 2023

Materai 6.000

Ferry Syarif Fuddin
NIM. 2120190398

HALAMAN PENGESAHAN

Nama : Ferry Syarif Fuddin
NIM : 2120190398
Judul : DEEP LEARNING MENGGUNAKAN TENSORFLOW DAN CONVOLUTIONAL NEURAL NETWORK UNTUK PENDETEKSIAN KEMASAN BISKUIT NEXTAR YANG RUSAK.

Telah dipertahankan di hadapan penguji pada tanggal Juli 2023.

Dewan Penguji Tim Pembimbing
Ketua Pembimbing I

..... **M. Jauhar Vikri, M. Kom**

NIDN: NIDN: 071208803

Anggota Pembimbing II

..... **Fetrika Anggraini, M.Pd.**

NIDN: NIDN: 0718038803

Mengetahui,
Dekan Fakultas Sains dan Teknologi Mengetahui
Ketua Program Studi

Sunu Wahyudhi, M.Pd.

NIDN: 0709058902

M. Jauhar Vikri, M. Kom

NIDN: 071208801

DAFTAR ISI

PERNYATAAN.....	ii
HALAMAN PENGESAHAN.....	iii
DAFTAR ISI.....	iv
DAFTAR TABEL.....	vii
DAFTAR GAMBAR	viii
DAFTAR LAMPIRAN.....	ix
BAB I	10
PENDAHULUAN.....	10
1.1. Latar Belakang.....	10
1.2. Rumusan Masalah	11
1.3. Tujuan Penelitian.....	12
1.4. Manfaat.....	12
1.5. Batasan Masalah.....	13
BAB II.....	14
TINJAUAN PUSTAKA.....	14
2.1. Landasan Teori	14
2.1.1. <i>Deep Learning</i>	14
2.1.2. Citra Digital	14
2.1.3. <i>Tensorflow</i>	14
2.1.4. <i>Convulational Neural Network (CNN)</i>	15
2.1.5. <i>Library</i>	18
2.1.6. <i>Feature Extraction</i>	19
2.1.7. <i>Computer Vision</i>	20
2.1.8. <i>Image Classification</i>	20
2.1.9. <i>Semantic Segmentation</i>	20
2.1.10. <i>Object Detection</i>	21
2.1.11. <i>Instance Segmentation</i>	21
2.1.12. <i>Machine Learning</i>	21
2.1.13. <i>Supervised Learning</i>	22

2.1.14.	<i>Unsupervised Learning</i>	22
2.1.15.	<i>Semi-Supervised Learning</i>	22
2.1.16.	<i>Reinforcement Learning</i>	22
2.2.	Penelitian Relevan.....	23
2.2.1.	Jurnal oleh Setyaningsih, E. R., dan Edy, M. S. Tahun 2022.....	23
2.2.2.	Jurnal oleh Setiawan, DKk Tahun 2020	24
2.2.3.	Jurnal oleh Tiara Sari dan Haryatmi Tahun 2021.....	25
2.2.4.	Jurnal oleh Pham dan Chang Tahun 2023	25
2.2.5.	Jurnal oleh Listyalina, Dkk Tahun 2022.....	26
2.2.6.	Jurnal oleh Priyanti Tahun 2021	27
2.2.7.	Jurnal oleh Trisiawan dan Yuliza Tahun 2022	28
2.2.8.	Jurnal oleh Setiani Tahun 2020	28
2.2.9.	Jurnal oleh Harika Dkk Tahun 2022.....	29
2.2.10.	Jurnal oleh Valentina Dkk Tahun 2020	30
2.2.11.	Jurnal oleh Valentina Dkk Tahun 2020	31
BAB III.....		37
METODE PENELITIAN.....		37
3.1.	Metode Penelitian.....	37
3.1.1.	Alur Penelitian	37
3.1.2.	Pengumpulan Data.....	37
3.1.3.	<i>Preprocessing Data</i>	38
3.1.4.	Pembuatan Model	39
3.1.5.	Pengujian Model.....	40
3.1.6.	Penerapan Data	41
3.2.	Desain Sistem	42
3.2.1.	Halaman Masuk	42
3.2.2.	Halaman Tes Uji	43
3.3.	Perhitungan Penerapan Model.....	46
BAB IV		47
IMPLEMENTASI DAN PEMBAHASAN.....		47
4.1.	Implementasi	Error! Bookmark not defined.

4.1.1.	Langkah-langkah Uji Coba.....	Error! Bookmark not defined.
4.2.	Pembahasan	56
4.2.1.	Evaluasi Metrik.....	56
4.2.2.	Perhitungan dan Analisis	57
	BAB V.....	59
	KESIMPULAN	59
5.1.	Kesimpulan.....	59
5.2.	Saran	60
	DAFTAR PUSTAKA	61
	LAMPIRAN	63

DAFTAR TABEL

Tabel 2.1. Hasil Penelitian Relevan	32
Tabel 4.1. Hasil Percobaan dengan jumlah dataset yang berbeda	50

DAFTAR GAMBAR

Gambar 2.1. <i>Flowchart</i> Rancangan	17
Gambar 3.1. Alur Penelitian.....	37
Gambar 3.2. Tampilan Kemasan yang Bagus.....	38
Gambar 3.3. Tampilan Gambar yang Rusak.....	38
Gambar 3.4. <i>Preprocessing</i> Data Yang Diambil Untuk Di <i>Training</i>	39
Gambar 3.5. <i>Login Form</i>	43
Gambar 4.1. Halaman Utama <i>Website</i>	54
Gambar 4.2. Memasang Pustaka yang digunakan	51
Gambar 4.3. Kode Pada Google Collab	53
Gambar 4.4. Hasil Validasi dari <i>website</i>	55

DAFTAR LAMPIRAN

Lampiran 1. Gambar dataset kemasan bagus	63
Lampiran 2. Gambar kemasan rusak.....	63
Lampiran 3. Kode <i>frontend</i> utama	64

BAB I

PENDAHULUAN

1.1. Latar Belakang

Perkembangan teknologi pangan dapat mempengaruhi berbagai kemasan pangan. Pengemasan makanan memiliki arti penting bagi setiap produk makanan. Tujuan pengemasan pangan adalah untuk mengisolasi pangan dari kondisi lingkungan normal, sehingga pengemasan memegang peranan penting dalam menjaga kebersihan dan higienitas pangan (Rorong & Wilar, 2020). Kemasan yang rusak dapat mengurangi masa simpan produk, menyebabkan kerusakan pada produk dan membahayakan kesehatan konsumen. Kemasan makanan yang rusak dapat terjadi karena beberapa faktor. Namun, dengan semakin meningkatnya permintaan produk makanan di pasar, terkadang sulit untuk mengecek setiap kemasan secara manual. Oleh karena itu, deteksi dini dan cepat terhadap kemasan makanan yang rusak sangat penting.

Deep learning memiliki kemampuan yang sangat baik dalam visi computer dengan kapabilitas nya yang signifikan dalam memodelkan berbagai data kompleks seperti data gambar. Salah satu metode *Deep learning* pada saat ini memiliki hasil paling bagus dalam pengenalan citra adalah *Convolutional Neural Network* (CNN). Hal itu disebabkan karena metode CNN berusaha meniru sistem pengenalan citra pada *visual cortex* manusia sehingga memiliki kemampuan mengolah informasi citra (Harani *et al.*, 2019). Pemanfaatan teknologi ini dapat diterapkan untuk mendeteksi kemasan makanan yang rusak dengan akurasi yang tinggi. Dengan kemampuannya untuk "mempelajari" fitur-fitur dari gambar secara otomatis.

TensorFlow merupakan kerangka kerja komputasi untuk membangun model pembelajaran mesin. *TensorFlow* menyediakan berbagai *toolkit* yang memungkinkan untuk membuat model pada tingkat abstraksi yang disukai dan dapat menjalankan grafik pada beberapa *platform hardware*, termasuk CPU, GPU, dan TPU (Hikmatia A.E & Ihsan Zul, 2021). *TensorFlow* menyediakan berbagai fitur dan alat yang memungkinkan pengguna untuk membangun dan

melatih model *Deep Learning* dengan lebih mudah dan efisien. Selain itu, *TensorFlow* juga dapat berjalan pada berbagai *platform* seperti *desktop*, *mobile*, dan *cloud*. Dalam proyek skripsi ini, akan digunakan *TensorFlow* untuk membangun metode CNN yang akan dilatih dengan menggunakan data gambar kemasan makanan. Dengan memanfaatkan kekuatan komputasi GPU dan teknologi pararel dari *TensorFlow* dengan metode CNN dapat dilatih dengan waktu yang lebih cepat dan menghasilkan akurasi yang lebih baik.

Dalam proyek skripsi ini, akan dibangun sebuah sistem deteksi kemasan makanan yang rusak menggunakan teknologi *Deep Learning* dengan menggunakan *TensorFlow* dan metode CNN. Data gambar kemasan makanan yang rusak dan tidak rusak akan digunakan untuk melatih model dan melakukan evaluasi. Hasil dari penelitian ini diharapkan dapat membantu industri makanan dalam meningkatkan kualitas produk dan mengurangi kerugian akibat kemasan makanan yang rusak.

1.2. Rumusan Masalah

Mengidentifikasi masalah ialah peneliti melakukan tahap pertama dalam melakukan penelitian, yaitu merumuskan masalah yang akan diteliti. Tahap ini merupakan tahap yang paling penting dalam penelitian, karena semua jalannya penelitian akan dituntun oleh perumusan masalah. Tanpa perumusan masalah yang jelas, maka peneliti akan kehilangan arah dalam melakukan penelitian (Ridha, 2017). Berdasarkan penjelasan tersebut, maka rumusan masalah pada penelitian ini adalah:

- 1.2.1. Bagaimana mempersiapkan dan memproses data citra kemasan makanan yang rusak agar dapat digunakan dalam pelatihan model CNN?
- 1.2.2. Bagaimana melakukan pelatihan model CNN menggunakan *TensorFlow* untuk mendeteksi kemasan makanan yang rusak dengan tingkat akurasi yang tinggi?
- 1.2.3. Bagaimana melakukan evaluasi dan validasi terhadap model CNN yang telah dilatih untuk mendeteksi kemasan makanan yang rusak?
- 1.2.4. Bagaimana mengimplementasikan model CNN yang telah dilatih ke dalam sistem pendekripsi kemasan makanan yang rusak secara *real-time*?

1.3. Tujuan Penelitian

Tujuan penelitian merupakan ungkapan sasaran yang akan dicapai dalam suatu penelitian. Tujuan penelitian harus dinyatakan dengan kongkrit, jelas dan ringkas dan dinyatakan dalam bentuk kalimat pernyataan. Isi dan rumusan tujuan penelitian harus mengacu pada rumusan masalah penelitian (Sugiono, 2019). dengan penjelasan tersebut, maka tujuan penelitian ini adalah:

- 1.3.1. Mengetahui persiapan dan proses data citra kemasan makanan yang rusak agar dapat digunakan dalam pelatihan model CNN.
- 1.3.2. Mengetahui pelatihan model CNN menggunakan *TensorFlow* untuk mendeteksi kemasan makanan yang rusak dengan tingkat akurasi tinggi.
- 1.3.3. Mengetahui evaluasi dan validasi terhadap model CNN yang telah dilatih untuk mendeteksi kemasan makanan yang rusak.

1.4. Manfaat

Manfaat hasil penelitian merupakan dampak dari tercapainya tujuan dan terjawabnya masalah yang telah dirumuskan (Sugiono, 2019). Maka dari itu manfaat dari penelitian ini adalah:

- 1.4.1. Meningkatkan keamanan pangan: Dengan menggunakan teknologi deep learning dan CNN untuk mendeteksi kemasan makanan yang rusak, dapat membantu mengurangi risiko konsumsi makanan yang tidak sehat atau berbahaya bagi kesehatan manusia.
- 1.4.2. Inovasi teknologi: Penelitian ini dapat memberikan kontribusi pada pengembangan teknologi pendekripsi kemasan makanan yang rusak dan mendorong inovasi teknologi di bidang pengolahan makanan.
- 1.4.3. Kontribusi ilmiah: Skripsi ini dapat memberikan kontribusi pada ilmu komputer dan pengolahan citra dalam penggunaan teknologi *deep learning* menggunakan *Tensorflow* dan CNN untuk deteksi kemasan makanan yang rusak.

1.5. Batasan Masalah

Berdasarkan latar belakang masalah dan identifikasi masalah di atas maka akan dilakukan pembatasan masalah yang diteliti sebagai berikut :

- 1.5.1. Penelitian ini akan memfokuskan pada pendekripsi kemasan biskuit merek Nextar yang rusak dan bagus. Jenis kemasan lainnya tidak akan diikutsertakan dalam penelitian ini.
- 1.5.2. Penelitian ini difokuskan untuk mengembangkan aplikasi agar dapat membantu industri pangan agar mampu mendekripsi kemasan Nextar yang rusak.
- 1.5.3. Penelitian menggunakan algoritma *Convolutional Neural Network* dengan penerapan dalam industry
- 1.5.4. Pengambilan dataset dari lingkungan berbasis industri dapat menghadapi keterbatasan akses data, seperti intensitas cahaya yang berbeda atau kendala lain yang mungkin mempengaruhi keberagaman dataset yang tersedia.

BAB II

TINJAUAN PUSTAKA

2.1. Landasan Teori

2.1.1. *Deep Learning*

Deep learning adalah bagian dari kecerdasan buatan dan *machine learning* yang merupakan pengembangan dari *neural network multiple layer* untuk memberikan ketepatan tugas seperti deteksi objek, pengenalan suara, terjemahan bahasa dan lainnya (Raup et al., 2022). *Deep learning* merupakan metode *learning* yang memanfaatkan *artificial neural network* yang berlapis-lapis(*multi layer*), *artificial neural network* ini dibuat mirip otak manusia, dimana neuron-neuron terkoneksi satu sama lain sehingga memberntuk sebuah jaringan neuron yang sangat rumit. *Deep learning* merupakan metode *learning* yang memanfaatkan *multiple non-linier transformation*, *deep learning* dapat dipandang sebagai gabungan dari *machine learning* dengan *artificial neural network*.

2.1.2. Citra Digital

Citra digital merupakan gambaran atau representasi digital dari objek citra yang tidak dapat dipisahkan dari kebutuhan manusia, semua objek telah diwujudkan memorinya dalam bentuk citra agar objek tersebut tidak pudar dimakan waktu (Ihsan et al., 2023). Secara umumnya, tampilan dialam semesta ini memiliki warna unik yang berbeda dan pastinya tidak hanya dalam bentuk warna putih dan hitam saja.

2.1.3. *Tensorflow*

Tensorflow adalah sistem *machine learning* yang beroperasi pada skala besar dan lingkungan yang heterogen (Fiddiyansyah et al., 2023). *Tensorflow* menggunakan grafik aliran data untuk mewakili komputasi, status bersama, dan operasi yang mengubah status tersebut. *Tensorflow* merupakan salah satu perangkat lunak (*software library*) *open-source* yang dikembangkan oleh *Google*

Brain Team untuk membangun dan melatih model *deep learning*. *Tensorflow* memiliki arsitektur yang *fleksibel* dan memungkinkan pengguna untuk membuat dan melatih model *deep learning* dengan berbagai jenis arsitektur dan data yang berbeda.

2.1.4. Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) adalah salah satu jenis jaringan saraf tiruan (*artificial neural network*) yang khusus digunakan untuk memproses data gambar dan video. CNN terdiri dari beberapa lapisan (*layer*) yang masing-masing memiliki peran dan fungsi yang berbeda dalam proses pengenalan pola dan fitur pada data gambar (Lesmana *et al.*, 2022).

Beberapa konsep penting dalam CNN adalah sebagai berikut:

2.1.4.1. Multinasional Layer

Convolutional Layer merupakan lapisan pertama pada CNN, yang bertugas untuk melakukan operasi konvolusi pada data gambar. Operasi konvolusi dilakukan dengan menggunakan *filter* atau *kernel* yang akan digeser pada seluruh area gambar. Setiap area yang dilalui oleh *filter* akan menghasilkan nilai konvolusi yang kemudian akan dijadikan fitur pada lapisan berikutnya.

2.1.4.2. Pooling Layer

Pooling Layer bertugas untuk mengurangi dimensi pada data gambar dengan melakukan operasi *pooling*, seperti *max pooling* atau *average pooling*. Operasi ini dilakukan dengan memilih nilai terbesar atau rata-rata pada area tertentu dari data gambar. Tujuannya adalah untuk mengurangi jumlah parameter pada model dan mempercepat proses *training*.

2.1.4.3. Fully-Connected Layer

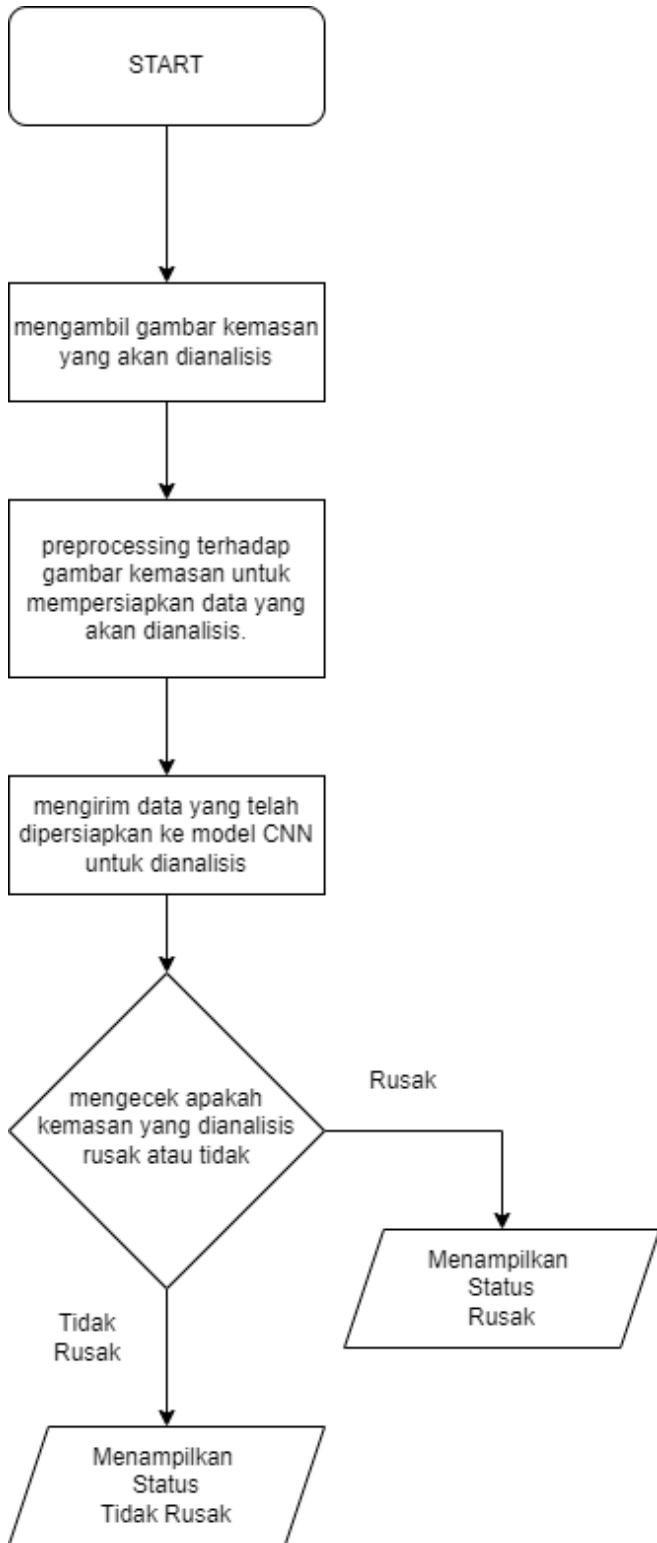
Fully-Connected Layer adalah lapisan terakhir pada CNN yang bertugas untuk menghubungkan fitur-fitur yang sudah dihasilkan oleh lapisan sebelumnya ke dalam kelas-kelas atau label-label yang sesuai dengan data gambar yang diberikan. Lapisan ini menggunakan algoritma seperti *softmax* atau *sigmoid* untuk menghasilkan probabilitas pada setiap kelas.

2.1.4.4. Activation Function

Activation Function adalah fungsi matematika yang digunakan untuk menambahkan *non-linearitas* pada model CNN. Fungsi ini diterapkan pada setiap neuron pada lapisan CNN, sehingga memungkinkan model untuk mempelajari hubungan *non-linear* antara fitur-fitur pada data gambar.

2.1.4.5. Loss Function

Loss Function adalah fungsi matematika yang digunakan untuk menghitung kesalahan (*error*) antara hasil prediksi dan label sebenarnya pada data gambar. *Loss Function* ini akan menjadi acuan bagi model untuk mengoptimalkan parameter pada setiap lapisan, sehingga dapat memperbaiki performa model pada setiap literasi *training*.



Gambar 2.1. Flowchart Rancangan

2.1.5. *Library*

Dalam penelitian yang dilakukan ini menggunakan *python* sebagai pengolah data yang dikirimkan oleh pengguna. Berikut *library* yang digunakan dalam penelitian :

2.1.5.1. *Sys*

Library ini menyediakan akses ke beberapa variabel dan fungsi yang mengatur perilaku *interpreter Python*. Digunakan untuk mengakses argumen baris perintah, mengelola jalur pencarian Python, dan beberapa fungsi *utilitas* lainnya.

2.1.5.2. *Os*

Library ini menyediakan akses ke berbagai fungsi yang digunakan untuk mengakses fungsi-fungsi sistem operasi seperti mengubah direktori kerja, membuat direktori, dan berinteraksi dengan file.

2.1.5.3. *IO*

Library ini menyediakan kelas utilitas untuk berinteraksi dengan input/output (I/O) data. Anda dapat menggunakan kelas-kelas di library ini untuk membaca atau menulis data dari berbagai sumber, termasuk data *string*, *file*, dan data *byte*.

2.1.5.4. *Re*

Library ini menyediakan ekspresi reguler (*regular expressions*) untuk mencocokkan dan memanipulasi teks. Anda dapat menggunakan library ini untuk mencari pola teks, melakukan penggantian teks, atau melakukan manipulasi teks lainnya.

2.1.5.5. *Base64*

Library ini menyediakan fungsi untuk melakukan *encoding* dan *decoding* data dalam bentuk *base64*. *Base64* adalah *representasi* data biner dalam bentuk teks ASCII agar dapat dengan mudah ditransmisikan melalui protokol yang hanya mendukung teks.

2.1.5.6. *Threading*

Library ini menyediakan dukungan untuk threading atau multithreading di *Python*. Fungsi *library* ini adalah membuat dan mengendalikan *thread* yang berjalan secara paralel untuk menjalankan tugas-tugas tertentu.

2.1.5.7. *PIL*

Sekarang dikenal sebagai *Pillow*: *Library* ini menyediakan berbagai fungsi untuk memanipulasi gambar dalam *Python*. *Library* ini digunakan untuk membuka, mengubah ukuran, menyimpan, dan memanipulasi gambar.

2.1.5.8. *Flask*

Flask adalah *framework* web ringan untuk bahasa pemrograman *Python* yang digunakan untuk membangun aplikasi web. Dengan *Flask*, dan digunakan untuk membuat *endpoint-endpoint* yang merespon permintaan *HTTP* dari pengguna.

2.1.5.9. *Pyngrok*

Library ini adalah pustaka *Python* untuk berkomunikasi dengan *ngrok*, alat yang memungkinkan untuk membuka jaringan lokal ke internet sehingga dapat mengakses server lokal dari alamat *URL* publik yang diberikan oleh *ngrok*.

2.1.5.10. *Flask_cors*

Library ini adalah ekstensi *Flask* yang menyediakan dukungan untuk *Cross-Origin Resource Sharing (CORS)* dalam aplikasi web. CORS memungkinkan sumber daya di halaman web untuk dimuat dari domain lain, yang diperlukan dalam beberapa kasus, misalnya saat memanggil API dari domain yang berbeda.

2.1.6. *Feature Extraction*

Feature extraction (ekstraksi ciri) merupakan proses untuk mengurangi jumlah sumber daya yang dibutuhkan untuk mendeskripsikan sebuah kumpulan data yang besar. Di dalam *computer vision* dan *image processing*, *feature* (ciri)

adalah potongan informasi tentang isi dari sebuah citra, dapat berupa titik, garis, atau objek. Hasil dari *ekstraksi ciri* dikenal dengan sebutan *feature vector* (vector ciri).

2.1.7. Computer Vision

Computer vision adalah bidang ilmiah yang membahas bagaimana komputer dapat memperoleh pemahaman tingkat tinggi dari gambar atau video digital. Dari perspektif teknik, *computer vision* berusaha untuk memahami dan mengotomatisasi tugas-tugas yang dapat dilakukan oleh sistem visual manusia. Tugas dari *computer vision* mencakup metode untuk memperoleh, memproses, menganalisis dan memahami citra digital, dan mengekstraksi data untuk menghasilkan informasi numerik atau simbolis.

2.1.8. Image Classification

Image classification adalah tugas dasar dari *computer vision* untuk mengkategorikan suatu gambar kedalam suatu kategori dari kumpulan kategori-kategori (misalnya, kucing, anjing, mobil, pesawat). Tujuannya adalah agar komputer dapat mengenali objek yang ada pada gambar. Metode yang digunakan untuk tugas klasifikasi diantaranya *Support Vector Machine* (SVM), *Decision Tree*, *K-nearest Neighbor*, *Artificial Neural Network* (ANN) dan *Convolutional Neural Network* (CNN).

2.1.9. Semantic Segmentation

Semantic segmentation mengacu pada proses yang menghubungkan setiap piksel pada gambar kedalam suatu label kelas yang sesuai, seperti klasifikasi gambar namun pada tingkat piksel. *Semantic segmentation* tidak memisahkan *instance* dari kelas yang sama, ini berarti bahwa jika ada dua objek dari kategori yang sama pada gambar maka peta segmentasi tidak secara inheren membedakannya sebagai objek yang terpisah. *Semantic segmentation* digunakan pada robotika, *autonomous driving*, agrikultur, dan diagnosa citra medis. Beberapa metode *deep learning* yang populer adalah *Fully Convolutional Network* (FCN), *U-Net*, *Mask R-CNN*, dan *DeepLab*.

2.1.10. Object Detection

Object detection (deteksi objek) adalah salah satu tugas penting pada *computer vision* yang berhubungan dengan pendekripsi objek visual dari kelas tertentu (seperti manusia, hewan, atau mobil) dalam citra digital. *Object detection* juga merupakan dasar untuk tugas *computer vision* lainnya, seperti *instance segmentation*, *image captioning*, *object tracking*, dan lain-lain. Kebanyakan algoritma yang dikembangkan sekarang ini adalah deteksi objek berbasis *deep learning*. Jenis pembuatan arsitektur deteksi objek dibagi menjadi dua kategori, *two-stage detector* dan *one-stage detector*. Kerangka *two-stage detector* membagi proses menjadi tahap *region proposal* dan tahap klasifikasi, Faster R-CNN adalah salah satu yang paling terkenal. Di sisi lain, *one-stage detector* dalam melakukan prediksi lokasi dan klasifikasi dilakukan dalam satu tahap sekaligus, YOLO dan SSD menjadi arsitektur yang terkenal (Manuel et al, 2021).

2.1.11. Instance Segmentation

Instance segmentation merupakan gabungan tugas dari *semantic segmentation* dan *object detection*. Selain mendekripsi objek, *instance segmentation* juga memprediksi setiap piksel yang membentuk objek tersebut. Tujuan dari *instance segmentation* adalah untuk mendapatkan area objek dari kelas yang sama, dibagi menjadi *instance* yang berbeda. Arsitektur model yang popular adalah Mask R-CNN.

2.1.12. Machine Learning

Menurut (Andriy, 2019) *machine learning* adalah sub bidang ilmu komputer yang berkaitan dengan pembuatan algoritma yang mengandalkan kumpulan contoh dari beberapa fenomena. Fenomena dapat berasal dari alam, buatan tangan manusia atau algoritma yang lain. *Machine learning* dapat didefinisikan sebagai proses untuk menyelesaikan sebuah permasalahan dengan mengumpulkan kumpulan data dan secara algoritma membangun model statistik berdasarkan kumpulan data tersebut. Terdapat beberapa istilah pembelajaran yang meliputi *supervised*, *semi-supervised*, *unsupervised*, dan *reinforcement*.

2.1.13. Supervised Learning

Pada tipe *supervised learning* dataset yang digunakan adalah kumpulan data yang telah diberi label. Tujuan dari *supervised learning* adalah menggunakan dataset untuk menghasilkan model yang mengambil vektor fitur sebagai informasi *input* dan *output* yang memungkinkan menyimpulkan label untuk vektor fitur tersebut.

2.1.14. Unsupervised Learning

Pada tipe *unsupervised learning* dataset yang digunakan adalah kumpulan data yang tidak diberi label. Tujuan dari *unsupervised learning* adalah untuk membuat model yang mengambil vektor fitur sebagai *input* dan mengubahnya menjadi vektor atau nilai lain yang dapat digunakan untuk memecahkan masalah praktis.

2.1.15. Semi-Supervised Learning

Pada tipe *semi-supervised learning* dataset yang digunakan terdiri dari data yang sudah diberi label dan yang tidak diberi label. Biasanya, jumlah dari data yang tidak diberi label lebih tinggi dibandingkan data yang diberi label. Tujuan *semi-supervised learning* sama dengan tujuan dari tipe *supervised learning*.

2.1.16. Reinforcement Learning

Pada *reinforcement learning* adalah subbidang dari *machine learning* di mana mesin “hidup” di dalam sebuah lingkungan dan mampu memahami keadaan lingkungan itu sebagai vektor fitur. Mesin dapat menjalankan aksi di setiap keadaan. Aksi yang berbeda dapat membawa *reward* yang berbeda dan juga dapat memindahkan mesin ke kondisi lingkungan yang lain. Tujuan dari algoritma *reinforcement learning* adalah untuk mempelajari kebijakan.

2.2. Penelitian Relevan

2.2.1. Jurnal oleh Setyaningsih, E. R., dan Edy, M. S. Tahun 2022

Berdasarkan jurnal ilmiah yang judul *YOLOv4* dan *mask R-CNN* untuk deteksi kerusakan pada karung (Setyaningsih & Edy, 2022). Penelitian ini bertujuan untuk mengembangkan model deteksi kerusakan pada karung komoditi menggunakan dua model *deep learning*, yaitu *YOLOv4* dan *Mask R-CNN*. Penulis menggunakan *dataset* yang terdiri dari gambar karung komoditi dengan kerusakan seperti robek, bolong, dan bercak. Berikut adalah tahapan-tahapan training model:

- Pra-pemrosesan *dataset*: Data diatur menjadi bentuk yang sesuai dengan model *YOLOv4*.
- Pelatihan model: Model *YOLOv4* dilatih dengan data pelatihan menggunakan algoritma *Stochastic Gradient Descent* (SGD) dengan *learning rate* sebesar 0,001, momentum sebesar 0,9, dan *weight decay* sebesar 0,0005. Pelatihan dilakukan selama 5000 *epoch* dengan *batch size* sebesar 4.
- Evaluasi model: Model *YOLOv4* dievaluasi menggunakan metrik *Average Precision* (AP) dengan IoU (*Intersection over Union*) *threshold* sebesar 0,5.

Hasil eksperimen menunjukkan bahwa kedua model *deep learning* dapat digunakan untuk deteksi kerusakan pada karung komoditi. Model *YOLOv4* mencapai akurasi deteksi sebesar 96,71%, sementara Model *Mask R-CNN* mencapai akurasi deteksi sebesar 97,06%. Namun, Model *Mask R-CNN* memerlukan waktu komputasi yang lebih lama dibandingkan dengan Model *YOLOv4*. Studi ini memberikan kontribusi dalam pengembangan teknologi deteksi kerusakan pada karung komoditi yang dapat membantu meningkatkan kualitas produk dan efisiensi produksi di industri. Selain itu, studi ini juga menunjukkan perbandingan antara dua model *deep learning* yang populer dalam deteksi objek, yaitu *YOLOv4* dan *Mask R-CNN*, yang dapat membantu peneliti dan praktisi dalam memilih model yang tepat untuk tugas deteksi objek yang serupa.

2.2.2. Jurnal oleh Setiawan, DKk Tahun 2020

Berjudul pendeksi kerusakan kemasan makanan menggunakan *tensorflow* dan *convolutional neural network* (Setiawan *et al.*, 2020). Dalam penelitian terdapat 3 tahapan utama dalam pengembangan model deteksi dan klasifikasi cacat pada kemasan kaleng menggunakan *Convolutional Neural Network* (CNN), yaitu:

- Pengumpulan data dan *preprocessing*

Dataset yang digunakan terdiri dari 1200 gambar kemasan kaleng yang terdiri dari tiga jenis cacat, yaitu retak, penyok, dan keriput. Gambar-gambar tersebut diproses dengan melakukan augmentasi data seperti *flipping*, *rotation*, dan *zooming* untuk meningkatkan variasi data.

- *Training model CNN*

Model CNN dikembangkan dengan menggunakan arsitektur *Convolutional Neural Network* (CNN) yang terdiri dari 6 layer, yaitu *Convolutional Layer*, *Maxpooling Layer*, *Batch Normalization Layer*, *Dropout Layer*, *Flatten Layer*, dan *Dense Layer*. Model CNN tersebut kemudian dilatih menggunakan data latih yang telah diproses sebelumnya. Hasil dari pelatihan tersebut menunjukkan bahwa model CNN mampu mendekksi cacat pada kemasan kaleng dengan akurasi yang baik.

- *Evaluasi model CNN*

Untuk mengevaluasi kinerja model CNN yang telah dilatih, dilakukan pengujian pada data uji yang terpisah dari data latih. Hasil dari pengujian menunjukkan bahwa model CNN mampu mendekksi tiga jenis cacat pada kemasan kaleng dengan akurasi yang baik. Model CNN yang dikembangkan berhasil mencapai akurasi deteksi sebesar 98,33% untuk cacat retak, 97,50% untuk cacat penyok, dan 96,67% untuk cacat keriput.

Dalam kesimpulannya, penelitian ini menunjukkan bahwa CNN dapat digunakan sebagai alat untuk deteksi dan klasifikasi cacat pada kemasan kaleng dengan akurasi yang tinggi, sehingga dapat membantu dalam meningkatkan efisiensi dan kualitas produksi di industri makanan dan minuman. Dalam hal ini, penggunaan teknologi CNN dapat membantu mengurangi waktu dan biaya yang diperlukan untuk memeriksa setiap kemasan kaleng secara manual.

2.2.3. Jurnal oleh Tiara Sari dan Haryatmi Tahun 2021

Berdasarkan jurnah ilmiah yang diterbitkan tahun 2021 dengan judul penerapan *convolutional neural network deep learning* dalam pendekripsi citra biji jagung kering (TiaraSari & Haryatmi, 2021). dijelaskan mengenai penerapan *Convolutional Neural Network* (CNN) untuk pendekripsi citra biji jagung kering. Penelitian ini dilakukan dengan mengambil sampel citra biji jagung kering dan melakukan preprocessing pada citra tersebut. Setelah itu, dilakukan pelatihan model CNN dengan menggunakan data latih yang telah diproses sebelumnya.

Hasil penelitian menunjukkan bahwa model CNN yang dikembangkan dapat digunakan untuk mendekripsi citra biji jagung kering dengan akurasi yang cukup tinggi. Model CNN tersebut berhasil mencapai akurasi sebesar 98,70% dalam mengenali citra biji jagung kering. Selain itu, penelitian ini juga menunjukkan bahwa penggunaan teknologi CNN dapat membantu dalam meningkatkan efisiensi dalam pengolahan citra dan pendekripsi cacat pada produk. Dalam hal ini, CNN dapat membantu mengurangi waktu dan biaya yang diperlukan untuk memeriksa citra biji jagung secara manual. Secara keseluruhan, penelitian ini menunjukkan bahwa penggunaan teknologi CNN dapat memberikan manfaat yang besar dalam bidang pengolahan citra dan pendekripsi cacat pada produk, khususnya pada bidang pertanian dan pangan.

2.2.4. Jurnal oleh Pham dan Chang Tahun 2023

Berdasarkan jurnah ilmiah yang berjudul *a YOLO-based real-time packaging defect detection system* (Vu *et al.*, 2023). Menjelaskan tentang pengembangan sistem pendekripsi cacat kemasan *real-time* berbasis *YOLO* (*You Only Look Once*) menggunakan teknik *deep learning*. Penelitian ini bertujuan untuk mengatasi masalah deteksi cacat kemasan yang masih bergantung pada pengamatan manual, sehingga memerlukan waktu yang lama dan tidak efisien. Oleh karena itu, penulis mengembangkan sistem pendekripsi cacat kemasan yang dapat bekerja secara *real-time* dan efisien. Sistem pendekripsi yang dikembangkan menggunakan arsitektur *YOLOv3* untuk mendekripsi cacat pada kemasan secara *real-time*. *YOLOv3* merupakan salah satu teknik *deep learning* yang paling umum digunakan dalam pendekripsi objek karena kecepatan dan akurasi yang baik.

Pada penelitian ini, penulis mengumpulkan 7 jenis cacat pada kemasan berbahan karton dan aluminium foil sebagai sampel data latih.

Dengan data latih tersebut diolah dengan teknik augmentasi data untuk menghindari *overfitting* pada model CNN yang digunakan. Hasil evaluasi menunjukkan bahwa sistem pendekripsi cacat kemasan yang dikembangkan memiliki akurasi yang baik, dengan nilai rata-rata *F1 score* sebesar 0,87. Dalam pengujian sistem *real-time*, sistem pendekripsi cacat kemasan berhasil mendekripsi cacat pada kemasan dengan kecepatan 30 fps (*frame per second*), sehingga sistem dapat digunakan dalam industri yang membutuhkan pendekripsi cacat pada kemasan dengan kecepatan tinggi. Secara keseluruhan, penelitian ini menunjukkan bahwa pendekatan *YOLOv3* dalam pengembangan sistem pendekripsi cacat kemasan dapat memberikan hasil yang baik dalam deteksi cacat pada kemasan secara *real-time*. Sistem yang dikembangkan dapat membantu meningkatkan efisiensi dan akurasi dalam deteksi cacat pada kemasan, sehingga dapat mengurangi kerugian pada industri yang bergantung pada kemasan yang berkualitas baik.

2.2.5. Jurnal oleh Listyalina, Dkk Tahun 2022

Berdasarkan jurnal ilmiah yang memiliki judul *deep-RIC: plastic waste classification using deep learning and resin identification codes (RIC)* (Listyalina *et al.*, 2022). Penelitian ini bertujuan untuk membantu dalam pengelolaan sampah plastik dengan melakukan klasifikasi secara otomatis menggunakan teknik *deep learning* dan *RIC*. *RIC* adalah kode identifikasi resin yang digunakan pada produk plastik untuk memudahkan proses pengolahan Kembali. Sistem klasifikasi yang dikembangkan menggunakan arsitektur *deep learning* yaitu *Convolutional Neural Network* (CNN) dengan transfer *learning* menggunakan model *MobileNetV2* sebagai *pre-trained model*. Model tersebut dilatih menggunakan *dataset* sampah plastik dengan 6 kategori berbeda sesuai dengan *RIC*. *Dataset* yang digunakan dalam penelitian ini diperoleh dari pengambilan gambar langsung dari sampah plastik yang tersedia. Gambar yang diambil kemudian diberi label dengan *RIC* yang sesuai untuk klasifikasi.

Hasil evaluasi menunjukkan bahwa sistem klasifikasi yang dikembangkan memiliki tingkat akurasi yang baik, dengan nilai akurasi rata-rata sebesar 91,44%. Sistem ini dapat memproses sampah plastik dalam jumlah besar dengan cepat dan akurat. Penelitian ini menunjukkan bahwa *deep learning* dan *RIC* dapat digunakan sebagai solusi untuk mengatasi masalah pengelolaan sampah plastik. Dengan menggunakan sistem klasifikasi ini, pengelolaan sampah plastik dapat dilakukan dengan lebih efisien dan akurat. Selain itu, hasil penelitian ini dapat digunakan sebagai dasar untuk mengembangkan sistem klasifikasi sampah plastik yang lebih canggih di masa depan.

2.2.6. Jurnal oleh Priyanti Tahun 2021

Berdasarkan jurnal ilmiah yang berjudul deteksi bakteri pada produk makanan kemasan menggunakan algoritma *naïve bayes* (Priyanti, 2021). Penelitian ini bertujuan untuk membantu dalam pengendalian kualitas produk makanan kemasan dengan melakukan deteksi bakteri secara cepat dan akurat. Algoritma *Naïve Bayes* dipilih karena memiliki kemampuan untuk mengklasifikasikan data dengan cepat dan akurat. Penelitian ini menggunakan data dari pengujian mikrobiologi pada produk makanan kemasan. Data tersebut mencakup hasil pengujian bakteri pada produk makanan kemasan yang dibagi menjadi 2 kelas yaitu positif dan negatif. Data yang telah dikumpulkan kemudian diolah dan dilakukan *preprocessing* dengan menggunakan beberapa teknik seperti *filtering*, *smoothing*, dan *scaling*. Setelah itu, dilakukan pelatihan pada algoritma *Naïve Bayes* dengan menggunakan data yang telah diolah.

Hasil evaluasi menunjukkan bahwa algoritma *Naïve Bayes* dapat digunakan untuk mendeteksi bakteri pada produk makanan kemasan dengan tingkat akurasi yang baik, yaitu sebesar 90,62%. Dalam penelitian ini, *Naïve Bayes* berhasil mengklasifikasikan data produk makanan kemasan dengan baik dan dapat membantu dalam pengendalian kualitas produk makanan kemasan secara efektif. Penelitian ini menunjukkan bahwa algoritma *Naïve Bayes* dapat digunakan sebagai solusi untuk mendeteksi bakteri pada produk makanan kemasan dengan cepat dan akurat. Selain itu, hasil penelitian ini juga dapat

menjadi dasar untuk pengembangan sistem deteksi bakteri pada produk makanan kemasan yang lebih canggih di masa depan.

2.2.7. Jurnal oleh Trisiawan dan Yuliza Tahun 2022

Berdasarkan jurnah ilmiah yang berjudul penerapan *multi-label image classification* menggunakan *metode convolutional neural network (CNN)* untuk sortir botol minuman (Trisiawan & Yuliza, 2022). membahas tentang penerapan teknik *multi-label image classification* menggunakan *metode Convolutional Neural Network (CNN)* untuk memilah botol minuman secara otomatis. Penelitian ini bertujuan untuk mengembangkan sistem yang dapat memilah botol minuman berdasarkan jenis dan kualitasnya secara otomatis menggunakan *teknik multi-label image classification*. Dalam penelitian ini, botol minuman diambil gambar dengan kamera dan dilabeli berdasarkan jenis dan kualitasnya.

Penelitian ini menggunakan arsitektur CNN untuk membangun model *multi-label image classification*. Data yang digunakan dalam penelitian ini terdiri dari 1800 gambar botol minuman, yang dibagi menjadi 1500 gambar untuk data training dan 300 gambar untuk data testing. Hasil dari penelitian menunjukkan bahwa teknik *multi-label image classification* dengan menggunakan arsitektur CNN dapat digunakan untuk memilah botol minuman secara otomatis berdasarkan jenis dan kualitasnya. Model yang dibangun berhasil memilah botol minuman dengan akurasi sebesar 91%.

2.2.8. Jurnal oleh Setiani Tahun 2020

Berdasarkan jurnah ilmiah yang berjudul Implementasi *Convolutional Neural Network* dengan Arsitektur *ResNet50* untuk Identifikasi Jenis Sampah Plastik (Setiani, 2020). melakukan penelitian untuk mengembangkan sistem pengenalan sampah plastik menggunakan metode deep learning, khususnya Convolutional Neural Network (CNN) dengan arsitektur *ResNet50*. Tujuan dari penelitian ini adalah untuk mengidentifikasi jenis sampah plastik secara otomatis dengan menggunakan gambar sebagai input. Penelitian ini menggunakan *dataset* gambar sampah plastik yang terdiri dari enam jenis sampah, yaitu botol plastik, gelas plastik, kantong plastik, kotak plastik, sedotan plastik, dan tutup botol

plastik. *Dataset* tersebut dilatih menggunakan metode transfer learning dengan arsitektur *ResNet50* dan dicoba menggunakan beberapa metode augmentasi gambar untuk meningkatkan performa model.

Hasil dari penelitian ini menunjukkan bahwa metode CNN dengan arsitektur *ResNet50* dapat menghasilkan akurasi yang cukup tinggi dalam mengidentifikasi jenis sampah plastik. Hasil terbaik yang diperoleh mencapai akurasi 99,63% dengan menggunakan metode augmentasi gambar yaitu flip vertical dan flip horizontal. Dengan menggunakan teknologi ini, diharapkan dapat membantu dalam mengurangi masalah sampah plastik dan meningkatkan kualitas lingkungan.

2.2.9. Jurnal oleh Harika Dkk Tahun 2022

Berdasarkan jurnal ilmiah yang berjudul penerapan klasifikasi untuk kelayakan hasil produksi jam tangan dengan menggunakan algoritma *k-nearest neighbor* (Harika *et al.*, 2022). membahas tentang penerapan algoritma *k-Nearest Neighbor* (K-NN) dalam melakukan klasifikasi untuk menentukan kelayakan hasil produksi jam tangan. Metode klasifikasi digunakan untuk mengevaluasi hasil produksi jam tangan dari beberapa kriteria seperti kerapatan, kekuatan tali, kualitas jahitan, dan penampilan keseluruhan. Membahas tentang penerapan algoritma klasifikasi untuk menentukan kelayakan hasil produksi jam tangan menggunakan algoritma *K-Nearest Neighbor* (K-NN).

Metode yang digunakan dalam penelitian ini adalah pengambilan data dengan melakukan pengukuran pada komponen jam tangan, kemudian dilakukan pengolahan data menggunakan *software* MATLAB dan proses klasifikasi menggunakan algoritma K-NN. Pada proses pengolahan data, dilakukan tahapan preprocessing seperti *filtering* dan ekstraksi fitur dengan metode *Gray Level Co-occurrence Matrix* (GLCM) untuk memperoleh informasi penting dari citra jam tangan. Kemudian, dilakukan pengelompokan data dengan algoritma K-NN dengan menggunakan perhitungan jarak Euclidean. Hasil penelitian menunjukkan bahwa penggunaan algoritma K-NN pada proses klasifikasi kelayakan hasil produksi jam tangan menghasilkan akurasi sebesar 96,7%. Hal ini menunjukkan

bahwa metode klasifikasi menggunakan algoritma K-NN dapat diaplikasikan untuk menentukan kelayakan hasil produksi jam tangan secara efektif dan efisien.

2.2.10. Jurnal oleh Valentina Dkk Tahun 2020

Berdasarkan jurnal ilmiah yang berjudul pengenalan gambar botol plastik dan kaleng minuman menggunakan metode *convolutional neural network* (Valentina *et al.*, 2020). Metode CNN dipilih karena keandalannya dalam mengenali pola visual pada gambar dan telah banyak digunakan pada berbagai aplikasi pengenalan gambar. Pada penelitian ini, *dataset* yang digunakan terdiri dari 220 gambar botol plastik dan 220 gambar kaleng minuman. Tahap pertama dari penelitian ini adalah melakukan *preprocessing* pada *dataset* untuk memastikan kualitas gambar yang digunakan dalam proses pengenalan. Proses *preprocessing* meliputi *resizing* gambar menjadi ukuran yang sama dan melakukan augmentasi data untuk meningkatkan variasi pada *dataset*. Selanjutnya, dilakukan pelatihan model CNN menggunakan arsitektur yang telah ditentukan.

Arsitektur yang digunakan terdiri dari beberapa *layer* konvolusi dan *pooling*, serta *layer fully connected* pada bagian akhir. Pengujian dilakukan dengan membagi *dataset* menjadi data latih dan data uji dengan perbandingan 80:20. Hasil pengujian menunjukkan bahwa model yang dihasilkan dapat mengenali botol plastik dan kaleng minuman dengan akurasi sebesar 97,5%. Hal ini menunjukkan bahwa metode CNN dapat digunakan untuk mengenali botol plastik dan kaleng minuman dengan baik dan dapat diaplikasikan pada berbagai bidang yang memerlukan pengenalan objek pada gambar.

2.2.11. Jurnal oleh Valentina Dkk Tahun 2020

Hasil pendekalian jerawat menggunakan deep learning dinilai dapat berjalan dengan baik. Jerawat pada wajah dapat dideteksi dengan nilai mAP dari model sebesar 42,2% dan AR (average recall) sebesar 32%. Jika dibandingkan dengan anotasi yang dilakukan oleh dokter kulit secara manual, model pendekalian memperoleh nilai recall (sensitivity) sebesar 77,2% dan precision 70,3%.

Tabel 2.1. Hasil Penelitian Relevan

No.	Peneliti	Topik	Metode	Hasil
1.	Setyaningsih, E. R., & Edy, M. S. (2022)	<i>YOLOv4 dan Mask R-CNN Untuk Deteksi Kerusakan Pada Karung Komoditi.</i>	<i>YOLOv4 dan R-CNN Menggunakan algoritma Stochastic Gradient Descent (SGD).</i>	Model <i>YOLOv4</i> mencapai akurasi deteksi sebesar 96,71%, sementara Model <i>Mask R-CNN</i> mencapai akurasi deteksi sebesar 97,06%. Namun, Model <i>Mask R-CNN</i> memerlukan waktu komputasi yang lebih lama dibandingkan dengan Model <i>YOLOv4</i> .
2.	Setiawan, Mulyana, dan Fauzi (2020)	Pendeteksian kerusakan kemasan makanan menggunakan <i>Tensorflow</i> dan <i>Convolutional Neural Network</i> .	CNN dengan melakukan augmentasi data seperti <i>flipping</i> , <i>rotation</i> , dan <i>zooming</i> .	Hasil dari pengujian menunjukkan bahwa model CNN mampu mendeteksi tiga jenis cacat pada kemasan kaleng dengan akurasi yang baik. Model CNN yang dikembangkan berhasil mencapai akurasi deteksi sebesar 98,33% untuk cacat retak, 97,50% untuk cacat penyok, dan 96,67% untuk cacat keriput.
3.	Tiara Sari dan Haryatmi (2021)	Penerapan <i>convolutional neural network deep learning</i> dalam pendekslan citra biji jagung kering.	CNN.	Hasil penelitian menunjukkan bahwa model CNN yang dikembangkan dapat digunakan untuk mendeksi citra biji jagung kering dengan akurasi yang cukup tinggi. Model CNN tersebut berhasil mencapai akurasi sebesar 98,70% dalam mengenali citra biji jagung kering.

No.	Peneliti	Topik	Metode	Hasil
4.	Pham dan Chang (2023)	<i>A YOLO-based Real-time Packaging Defect Detection System.</i>	YOLOv3 dan CNN.	Hasil evaluasi menunjukkan bahwa sistem pendekripsi cacat kemasan yang dikembangkan memiliki akurasi yang baik, dengan nilai rata-rata F1 score sebesar 0,87. Dalam pengujian sistem <i>real-time</i> , sistem pendekripsi cacat kemasan berhasil mendekripsi cacat pada kemasan dengan kecepatan 30 fps (<i>frame per second</i>), sehingga sistem dapat digunakan dalam industri yang membutuhkan pendekripsi cacat pada kemasan dengan kecepatan tinggi.
5.	Listyalina, Yudianingsih, Soedjono, Utari, dan Dharmawan (2022)	<i>Deep-RIC: Plastic Waste Classification using Deep Learning and Resin Identification Codes (RIC).</i>	<i>Convolutional Neural Network (CNN)</i> dengan <i>transfer learning</i> menggunakan model <i>MobileNetV2</i> sebagai <i>pre-trained model</i> .	Hasil evaluasi menunjukkan bahwa sistem klasifikasi yang dikembangkan memiliki tingkat akurasi yang baik, dengan nilai akurasi rata-rata sebesar 91,44%. Sistem ini dapat memproses sampah plastik dalam jumlah besar dengan cepat dan akurat.
6.	Priyanti (2021)	Deteksi bakteri pada	Algoritma <i>Naïve Bayes</i>	Hasil evaluasi menunjukkan bahwa algoritma

No.	Peneliti	Topik	Metode	Hasil
		produk makanan kemasan menggunakan algoritma <i>Naïve Bayes</i> .	dan <i>preprocessing</i> menggunakan <i>filtering</i> , <i>smoothing</i> , dan <i>scaling</i> .	<i>Naïve Bayes</i> dapat digunakan untuk mendeteksi bakteri pada produk makanan kemasan dengan tingkat akurasi yang baik, yaitu sebesar 90,62%.
7.	Trisiawan, Yuliza, dan Attamimi (2022)	Penerapan <i>multi-label image classification</i> menggunakan metode <i>convolutional neural network</i> (CNN) untuk sortir botol minuman.	CNN.	Hasil dari penelitian menunjukkan bahwa teknik <i>multi-label image classification</i> dengan menggunakan arsitektur CNN dapat digunakan untuk memilah botol minuman secara otomatis berdasarkan jenis dan kualitasnya. Model yang dibangun berhasil memilah botol minuman dengan akurasi sebesar 91%.
8.	Setiani (2020)	Implementasi <i>Convolutional Neural Network</i> dengan Arsitektur <i>ResNet50</i> untuk Identifikasi Jenis Sampah Plastik.	CNN dengan dilatih menggunakan metode <i>transfer learning</i> dengan arsitektur <i>ResNet50</i> .	Hasil terbaik yang diperoleh mencapai akurasi 99,63% dengan menggunakan metode augmentasi gambar yaitu <i>flip vertical</i> dan <i>flip horizontal</i> .
9.	Harika, M., Ramdania, D.	Penerapan klasifikasi untuk kelayakan hasil	<i>Algoritma K-Nearest Neighbor</i> dengan	Hasil penelitian menunjukkan bahwa penggunaan algoritma K-NN pada proses klasifikasi kelayakan

No.	Peneliti	Topik	Metode	Hasil
	R., Hidayat, R. S., Oktarini, S., & Feirizal, F. (2022)	produksi jam tangan dengan menggunakan algoritma <i>K-Nearest Neighbor.</i>	metode <i>Gray Level Co- occurrence Matrix.</i>	hasil produksi jam tangan menghasilkan akurasi sebesar 96,7%.
10.	Valentina, R., Rostianingsih, S., & Tjondrowiguno, A. N. (2020)	Pengenalan gambar botol plastik dan kaleng minuman menggunakan metode <i>convolutional neural network.</i>	<i>Convolutional neural network</i> (CNN).	Hasil pengujian menunjukkan bahwa model yang dihasilkan dapat mengenali botol plastik dan kaleng minuman dengan akurasi sebesar 97,5%.

Berdasarkan studi pustaka yang dilakukan mengenai penggunaan *Deep Learning* dengan *Tensorflow* dan *Convolutional Neural Network (CNN)* untuk pendekalian kemasan makanan yang rusak, dapat disimpulkan bahwa:

- *Deep Learning* merupakan salah satu teknik *Machine Learning* yang mampu memproses data dalam jumlah besar dan kompleks dengan cepat dan akurat.
- *Tensorflow* adalah salah satu *platform* atau *library* yang populer digunakan dalam implementasi *Deep Learning* dan memudahkan pengguna dalam membuat dan mengoptimasi model *Deep Learning*.
- CNN merupakan salah satu jenis arsitektur *Deep Learning* yang efektif digunakan dalam tugas-tugas pengenalan citra atau *image recognition*.
- Pendekalian kemasan biskuit Nextar yang rusak dapat dilakukan dengan menggunakan teknik *Deep Learning* dengan *Tensorflow* dan CNN, dengan data set yang diolah melalui beberapa tahap *preprocessing*.
- Dalam penelitian-penelitian terdahulu, teknik *Deep Learning* dengan *Tensorflow* dan CNN telah terbukti berhasil dalam melakukan pendekalian cacat atau kerusakan pada produk makanan lainnya.

Dari kesimpulan tersebut, dapat disimpulkan bahwa teknik *Deep Learning* dengan *Tensorflow* dan CNN merupakan salah satu metode yang efektif dan dapat digunakan untuk pendekalian kemasan biskuit Nextar yang rusak, dengan asumsi pengumpulan dan *preprocessing* data yang tepat dilakukan.

BAB III

METODE PENELITIAN

3.1. Metode Penelitian

3.1.1. Alur Penelitian

Penelitian ini memiliki alur sebagai tahapan dalam pengembangan penelitian. Sebagai berikut:



Gambar 3.1. Alur Penelitian

Pada tahapan pertama melakukan pencarian dan pengambilan dataset dari beberapa sumber seperti internet dan foto secara langsung. Tahap kedua adalah penyesuaian ukuran gambar tersebut dengan mengubah ukuran gambar menjadi lebih kecil agar tidak memberatkan dataset saat dibaca . Tahapan ketiga yaitu melakukan konversi gambar ke grayscale. Tahapan keempat yaitu melakukan proses training dan testing. Tahapan kelima adalah membuat parameter

3.1.2. Pengumpulan Data

Data yang digunakan dalam penelitian ini adalah citra kemasan biskuit Nextar yang baik dan rusak. Data baik diambil dari sampel produksi yang dianggap baik dan data rusak diambil dari sampel produksi yang mengalami kerusakan kemasan. Data yang diperoleh akan dibersihkan dan diproses pada tahap berikutnya. Berikut Contoh sampel data yang diambil untuk diolah:



Gambar 3.2. Tampilan Kemasan yang Bagus



Gambar 3.3. Tampilan Gambar yang Rusak

3.1.3. *Preprocessing Data*

Pada tahap ini, data yang diperoleh akan dibersihkan dan diproses untuk menghilangkan *noise* dan memperbaiki kualitas citra. *Preprocessing* data meliputi normalisasi citra, serta *resizing* citra untuk menyesuaikan ukuran citra yang diinginkan. *Preprocessing* Data ini dilakukan agar tidak memperberat kinerja komputer yang digunakan karena data yang digunakan memiliki ukuran yang besar



Gambar 3.4. *Preprocessing Data Yang Diambil Untuk Di Training*

3.1.4. Pembuatan Model

Model *deep learning* akan dibangun menggunakan *tensorflow* dan CNN. Pembuatan model akan dilakukan dengan mengatur parameter seperti jumlah lapisan (*layers*), fungsi aktivasi, dan pengaturan lainnya. Model akan dilatih menggunakan data baik dan rusak yang telah dipreproses. Pada penelitian ini, model CNN yang digunakan terdiri dari beberapa lapisan, yaitu:

- Lapisan *Input*: berfungsi sebagai *input* data gambar.
- Lapisan *Konvolusi*: berfungsi untuk mengekstrak fitur-fitur dari gambar menggunakan kernel konvolusi.
- Lapisan *ReLU*: berfungsi untuk mempercepat proses pelatihan dengan menghilangkan nilai negatif pada lapisan konvolusi.

- Lapisan *Max Pooling*: berfungsi untuk mengurangi dimensi dari hasil konvolusi dan mempertahankan fitur yang paling penting.
- Lapisan *Dropout*: berfungsi untuk mencegah *overfitting* dengan menonaktifkan beberapa neuron secara acak selama pelatihan.
- Lapisan *Flatten*: berfungsi untuk meratakan hasil *pooling* menjadi vektor fitur.
- Lapisan *Dense*: berfungsi untuk menghubungkan vektor fitur dengan kelas *output*.
- Lapisan *Output*: berfungsi sebagai *output* hasil klasifikasi.

3.1.5. Pengujian Model

Model selesai dilatih/train akan dilakukan proses pengujian model. Dengan menggunakan *python* sebagai *platform* untuk mengevaluasi hasil model yang telah dilatih. Dengan menggunakan *library* seperti *NumPy* dan Pandas untuk membaca dan memuat dataset. Selanjutnya, dapat menggunakan *tensorflow* dan Keras untuk membangun model CNN dan melatih model dengan *dataset* yang dimuat agar dapat diteruskan untuk mengevaluasi performa model menggunakan metrik evaluasi seperti berikut:

3.1.5.1. Akurasi

Akurasi adalah ukuran persentase dari jumlah klasifikasi yang benar dibandingkan dengan total klasifikasi yang dilakukan. Akurasi dihitung dengan membagi jumlah klasifikasi yang benar dengan jumlah total klasifikasi.

3.1.5.2. Presisi

Presisi adalah ukuran persentase dari jumlah hasil klasifikasi yang benar positif dibandingkan dengan total hasil klasifikasi positif. Presisi dihitung dengan membagi jumlah hasil klasifikasi benar positif dengan jumlah total hasil klasifikasi positif.

3.1.5.3. Recall

Recall adalah ukuran persentase dari jumlah hasil klasifikasi yang benar positif dibandingkan dengan total jumlah data yang seharusnya diklasifikasikan positif. *Recall* dihitung dengan membagi jumlah hasil klasifikasi benar positif dengan jumlah total data yang seharusnya diklasifikasikan positif.

3.1.5.4. *FI-Score*

F1-score adalah rata-rata harmonik dari presisi dan *recall*. *F1-score* memberikan ukuran yang lebih seimbang antara presisi dan recall daripada hanya menggunakan satu dari dua metrik ini saja.

Dalam pengujian ini *output* dari *python* hanya sebatas data *output* label dan akurasi yang ditampilkan masih dalam bentuk data *float* dan memerlukan sebuah *library* lagi bernama *Flask* untuk membuat data uji menjadi *Backend Restful API* supaya bisa diolah lagi oleh *frontend website* agar mudah digunakan oleh pengguna. Agar aplikasi berjalan lancar dengan keamanan yang baik, maka menggunakan text kunci agar tidak semua orang bisa mengakses datanya.

3.1.6. Penerapan Data

Setelah selesai membuat *backend Rest API* untuk menguji data model maka dibutuhkan *frontend* agar pengujian model bisa dilakukan oleh *user* dengan nyaman. Dalam penelitian ini mengambil *framework javascript vue* sebagai *frontend*. *Vue* adalah salah satu *framework JavaScript* yang dapat digunakan untuk membangun aplikasi web interaktif dengan cepat dan mudah.

Dalam konteks penggunaannya sebagai *frontend* dari *deep learning* menggunakan *tensorflow* dan *convolutional neural network* untuk pendekripsi kemasan biskuit Nextar yang rusak, *Vue* dapat mempermudah penggunaan dan integrasi antara model *deep learning* dan tampilan aplikasi web. dalam proses pendekripsi kemasan biskuit nextar yang rusak, model *deep learning* yang dibuat menggunakan *tensorflow* dan *convolutional neural network* digunakan untuk memproses gambar kemasan biskuit dan menghasilkan prediksi apakah kemasan biskuit tersebut rusak atau tidak. Kemudian, hasil prediksi tersebut ditampilkan pada tampilan aplikasi web yang dibangun dengan *Vue*.

Menggunakan *Vue* sebagai *frontend*, pengguna dapat dengan mudah mengakses dan memanipulasi hasil prediksi dari model *deep learning*, serta menampilkan informasi tambahan seperti gambar kemasan biskuit yang diproses dan hasil prediksi yang dihasilkan. Hal ini dapat meningkatkan efektivitas dan efisiensi dalam proses pendekripsi kemasan biskuit yang rusak. Selain itu, *Vue* juga memiliki kemampuan untuk mengintegrasikan komponen-komponen lain

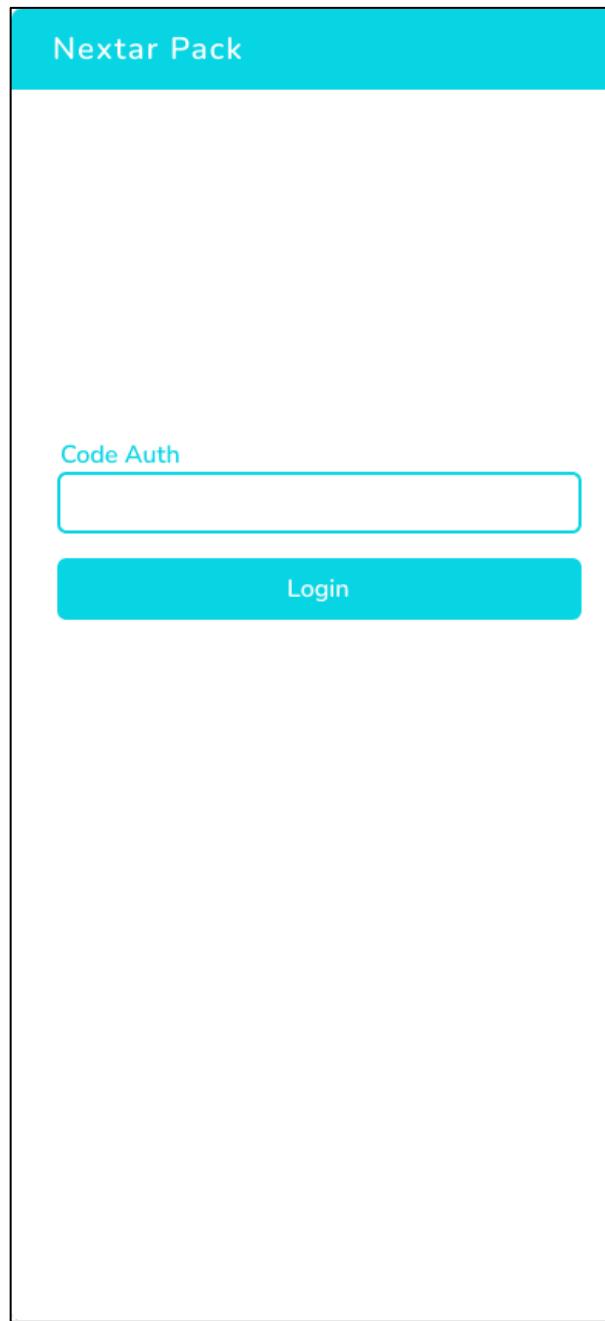
seperti grafik dan tabel, yang dapat membantu pengguna dalam memvisualisasikan dan menganalisis hasil prediksi dari model *deep learning* dengan lebih mudah dan cepat.

3.2. Desain Sistem

Dalam pengembangan penelitian ini dengan menggunakan sistem yang diperlukan beberapa halaman ini agar keamanan dan kenyamanan pengguna terlaksana. Struktur halamannya sebagai berikut :

3.2.1. Halaman Masuk

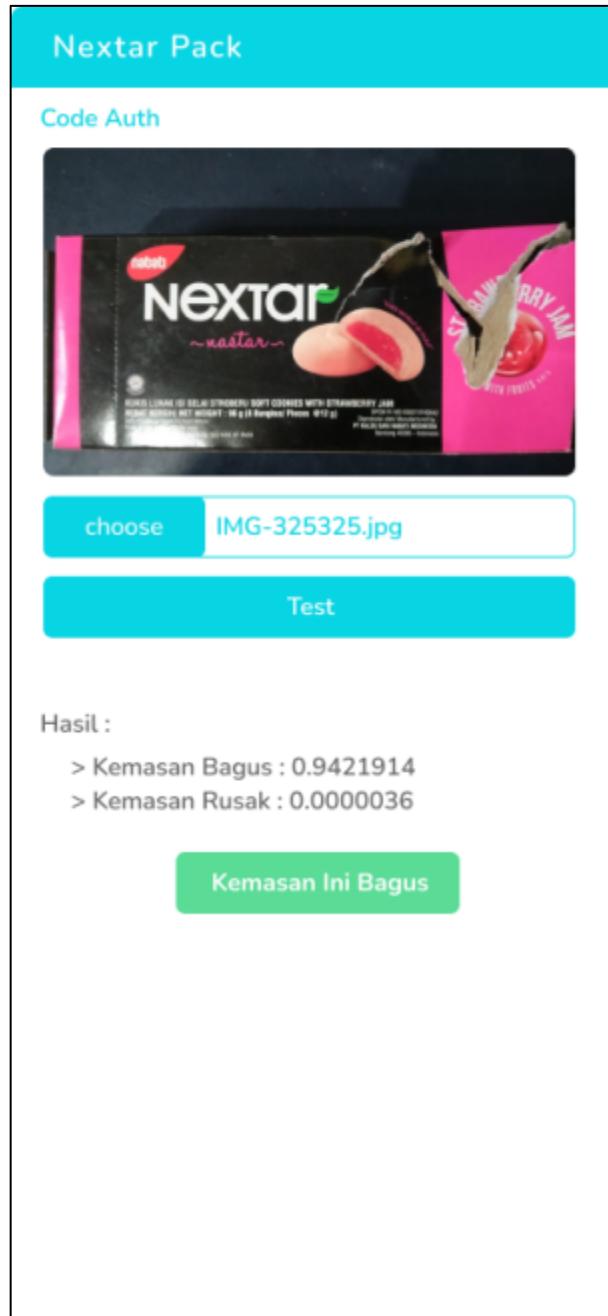
Dalam *Login Form* ini menggunakan *key* untuk dikirim kepada *backend python* nya agar mendapatkan autentikasi untuk mendapatkan akses data. *Login Form* ini berfungsi agar tidak semua orang dapat mengakses data.



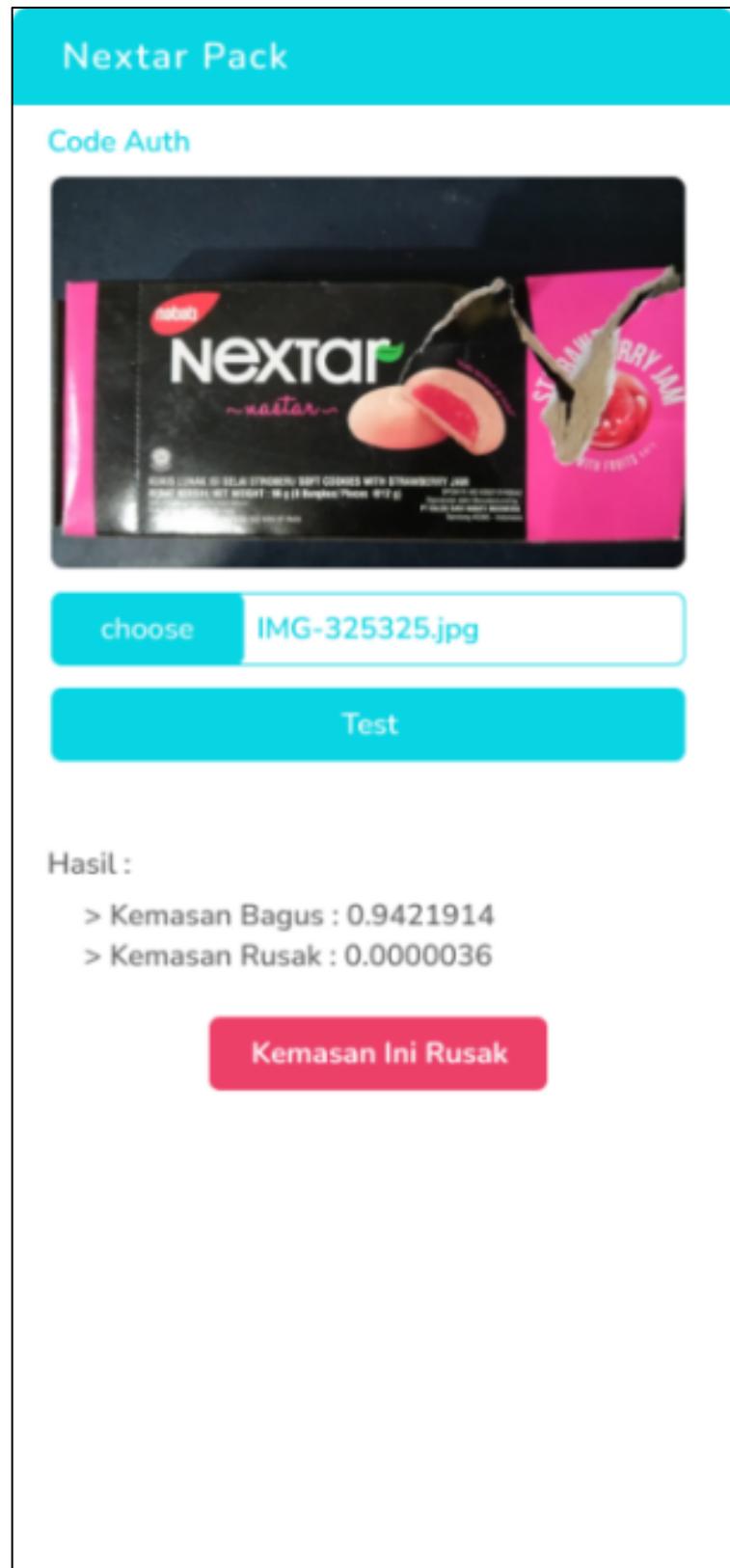
Gambar 3.5. Login Form

3.2.2. Halaman Tes Uji

Dalam halaman tes uji ini berisi mengenai hal hal yang diperlukan dalam melakukan tes uji. Ada gambar yang harus dikirim untuk dilakukan tes uji apakah gambar tersebut merupakan kemasan nextar yang bagus apa yang rusak.



Gambar 3.6. Kemasan Bagus



Gambar 3.6. Kemasan Bagus

3.3. Perhitungan Penerapan Model

Berikut ini adalah beberapa rumus perhitungan yang mungkin diperlukan dalam penerapan model *Deep Learning* menggunakan *TensorFlow* dan *Convolutional Neural Network* :

Convolutional layer:

- a. Jumlah filter: F
- b. Ukuran filter: K
- c. Ukuran input: N x N
- d. Ukuran output: M x M (dalam hal ini, $M = N - K + 1$)
- e. Rumus perhitungan jumlah parameter pada setiap *filter*: $(K \times K \times C) + 1$
- f. Rumus perhitungan jumlah parameter pada *layer*: $F \times ((K \times K \times C) + 1)$

Pooling layer:

- a. Ukuran filter: K
- b. Ukuran input: N x N
- c. Ukuran output: M x M (dalam hal ini, $M = N / K$)

Fully connected layer:

- a. Jumlah neuron: N
- b. Jumlah input: I
- c. Jumlah output: O
- d. Rumus perhitungan jumlah parameter: $(I+1) \times N + (N+1) \times O$

Backpropagation:

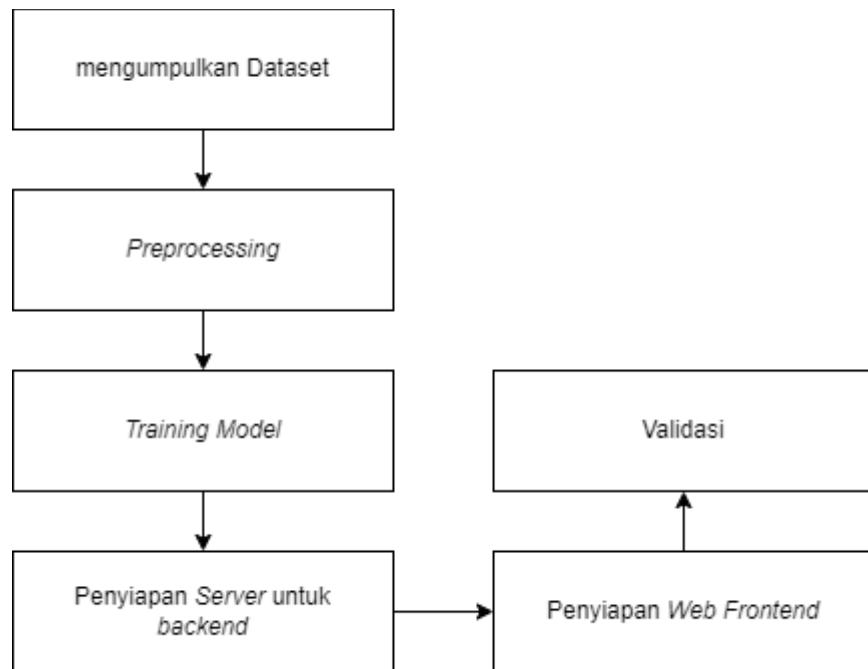
- a. *Loss function*: L
- b. *Gradient of Loss function*: ∇L
- c. *Learning rate*: α
- d. *Update weight*: $W_{new} = W_{old} - \alpha \times \nabla L$

BAB IV

IMPLEMENTASI DAN PEMBAHASAN

4.1. IMPLEMENTASI

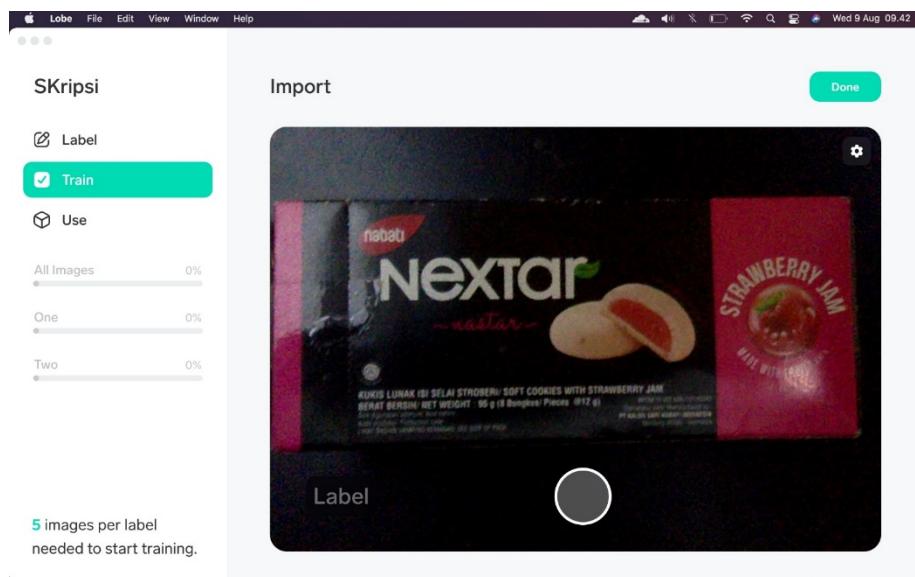
Mengangkat tentang implementasi *deep learning* dalam mengatasi permasalahan kemasan rusak pada kemasan Nextar. Dalam bab ini, akan dijelaskan bagaimana penerapan teknik-teknik *deep learning* untuk mendeteksi dan mendiagnosis kerusakan pada kemasan. Pada penelitian ini menggunakan aplikasi Lobe untuk membantu dalam melakukan pengumpulan dataset , preprocessing, dan training model. Lobe adalah aplikasi buatan tim Microsoft yang ditujukan membantu orang untuk menggunakan *model machine learning* dengan menggunakan desain *interface* yang mudah digunakan. Berikut gambar 4.1 alur dalam pengimplementasian penelitian ini.



Gambar 4.1 *Flowchart* pada Bab 4

4.1.1 Pengumpulan Dataset

Dataset gambar kemasan Nextar yang bagus dan rusak dengan rasa stroberi dikumpulkan dan dibuat oleh peneliti dengan bantuan aplikasi Lobe. Hanya gambar yang memenuhi kriteria yang diambil dari hasil pencarian. Sampel data gambar yang digunakan ditunjukkan pada Gambar 3.1 dan Gambar 3.2. Kriteria gambar yang digunakan adalah gambar yang hanya mencakup tampilan depan produk dan berukuran lebih dari 150 piksel. Berikut tampilan pengumpulan dataset menggunakan aplikasi Lobe.



4.1.2 Preprocessing

Sebelum memulai proses pelatihan model, preprocessing gambar dilakukan untuk mempersiapkan dataset. pengukuran gambar, dan labeling gambar adalah langkah-langkah yang dilakukan secara bertahap.

- Kompresi Gambar

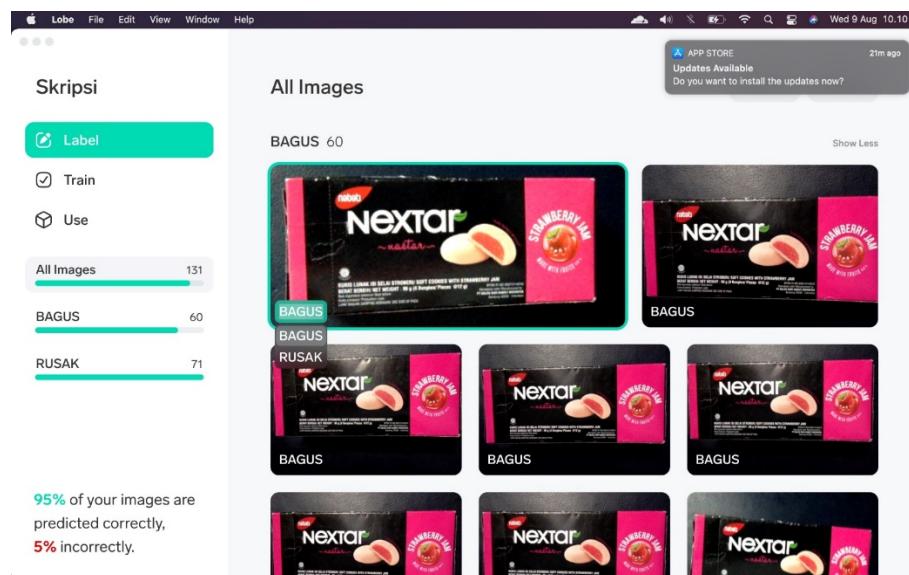
Standarisasi ukuran diperlukan karena ukuran citra input bervariasi.

Karena kompleksitas gambar yang terlalu besar, ukuran gambar yang terlalu besar dapat mengganggu proses komputasi, sedangkan ukuran gambar yang terlalu kecil dapat mempercepat proses, tetapi dapat menyebabkan informasi dari gambar hilang. Akibatnya, pengukuran

gambar disesuaikan dengan ukuran objek yang ingin dideteksi. Setelah gambar yang dihasilkan dari segmentasi kemasan dikompresi menjadi gambar dengan ukuran yang lebih kecil. Gambar dengan ukuran ini bukan gambar yang besar, jadi objek kemasan masih dapat dilihat secara visual.

- Labeling Gambar

Dengan menggunakan *software Lobe*, anotasi gambar atau label gambar dilakukan untuk mengidentifikasi objek yang ingin dideteksi pada gambar. Ini dilakukan dengan menambahkan teks atau label ke setiap bagian gambar data yang telah dikumpulkan. Label "Bagus" dan "Rusak" digunakan di sini.



4.1.3 Training Model

Model diberikan data pelatihan, dan *processing* dilakukan untuk mengubah bobot dan parameter model agar lebih sesuai dengan data yang akan dilakukan dalam pelatihan. Proses ini melibatkan perhitungan *loss* atau *error* antara output model dan label yang benar, dan kemudian melakukan penyesuaian bobot agar *error* ini diminimalkan. Berikut tabel hasil training yang dilakukan dengan bantuan aplikasi Lobe.

Jumlah Data	Waktu Pelatihan	<i>Correct</i>	<i>Loss / error</i>
30	~ 3 menit	90 %	10 %
80	~ 4 menit	89 %	11 %
130	~ 5 menit	95 %	5 %
200	~5 menit	100%	0%
411	~ 20 menit	100%	0%

Tabel 4.1. Hasil Percobaan dengan jumlah dataset yang berbeda

4.1.4 Penyiapan *Server Backend*

Setelah melakukan proses training pada aplikasi Lobe, pada proses selanjutnya adalah mengeksperimen hasil *model training* untuk dikonsumsi *library Tensorflow* pada kode *Python* yang digunakan untuk melakukan validasi gambar inputan dari pengguna. Kode Python ini berfungsi untuk memberikan respon data persentase dari kemiripan gambar yang dikirim pengguna melalui *web frontend* menggunakan *Restful Api*. Dalam proses *backend web* yang akan digunakan adalah menggunakan Google Collab. Google Collab dipilih menjadi opsi karena fitur spesifikasi *server* yang mampu untuk melakukan komputasi *image processing* karena membutuhkan *Graphic Processing Unit* dengan performa yang bagus. Selain itu digunakannya Google Collab ini karena ada fitur penyimpanan untuk menyimpan data *model training* yang akan digunakan, serta karena Google Collab gratis untuk digunakan. Berikut tahapan dalam penyiapan *server backend* menggunakan Google Collab.

a. Mengunggah Dataset ke Google Drive

Google Colab harus memiliki akses ke program dan dataset yang diperlukan. Selanjutnya, berkas diunggah ke Google Drive, yang dapat terhubung secara langsung dengan Google Colab. Ini dilakukan dengan mengunggah berkas ke folder yang sudah ditentukan untuk pelatihan model, atau workspace. Ini memudahkan proses mounting Google Drive ke Google Colab, sehingga tidak perlu mengubah alamat folder setiap kali diperlukan.

b. Menghubungkan Google Colab dan Google Drive

Untuk menghubungkan Google Collab dan Google Drive, pengguna harus memberikan hak akses untuk kedua layanan tersebut. Ini diperlukan saat proses mounting folder ke Google Colabs. Google akan meminta kode token otorisasi dari pengguna untuk memberikan izin untuk mengakses alamat Google Drive. Jika pengguna memberikan izin, Google akan memberikan kode hak akses untuk membiarkan Google Colabs mengakses berkas-berkas yang didaur ulang.

c. Instalasi *Library Python*

Lembar kerja Google Collab menggunakan bahasa pemrograman *Python*. Untuk menghindari *error* program, sebelum memulai proses pelatihan model, pastikan bahwa *library* yang digunakan sudah terinstal. Daftar *library Python* yang digunakan ditunjukkan pada Tabel 4.2 berikut.

```
[ ] !pip install flask
!pip install pillow
!pip install tensorflow
!pip install flask_cors
!pip install flask-ngrok
!pip install pyngrok
!ngrok authtoken '21o942uQaReHpKWqDj7w7QGmJ0q_pn1WFu248ro122B91gAf'
```

Gambar 4.1. Memasang Pustaka yang digunakan

<i>Flask</i>	Berguna untuk menghandel <i>request API</i> dari halaman utama <i>website</i>
<i>Pillow</i>	Digunakan untuk memanipulasi gambar
<i>Tensorflow</i>	<i>Library</i> untuk pelatihan model <i>deep learning</i>
<i>Flask_cors</i>	<i>Library</i> yang diperlukan supaya akses data

	dari luar <i>domain</i> yang dipakai dapat dijalankan tanpa terkena <i>blocking data</i>
<i>Flask- ngrok</i>	<i>Library</i> yang diperlukan untuk menjalankan layanan Ngrok supaya mendapat <i>public ip</i> agar bisa diakses diluar lingkungan lokal
<i>Pyngrok</i>	<i>Library</i> yang diperlukan untuk menjalankan layanan Ngrok supaya mendapat <i>public ip</i> agar bisa diakses diluar lingkungan lokal

Tabel 4.2. Tabel fungsi dari *library* yang digunakan

d. Menjalankan kode *python*

Kode *backend* yang berisi kode python untuk melakukan validasi gambar yang diupload lewat halaman *frontend*. Ada banyak opsi untuk menjalankan kode *backend* ini. Bisa lewat *Virtual Private Server* (VPS) untuk menjalankan, tetapi membutuhkan biaya yang cukup banyak untuk menyewa sebuah VPS. Jadi untuk alternatif agar bisa diakses secara publik bisa menggunakan Google Collabs.

```

import sys
import os
import io
import re
import base64
import threading

from PIL import Image
from flask import Flask, request
from flask_cors import cross_origin
from flask import Flask
from pyngrok import ngrok

# py_file_location = "/content/drive/MyDrive/skripsi_nextar/"
py_file_location = "/content/drive/MyDrive/skripsi_nextar_v2/"
sys.path.append(os.path.abspath(py_file_location))
from tf_model_helper import TFModel

# from flask_ngrok import run_with_ngrok
port = 5000
app = Flask(__name__)
public_url = ngrok.connect(port).public_url
print(" * ngrok tunnel \"{}\" -> \"http://127.0.0.1:{}\"".format(public_url, port))
app.config["BASE_URL"] = public_url
# run_with_ngrok(app)

# Path to signature.json and model file
ASSETS_PATH = os.path.join(".", "/content/drive/MyDrive/skripsi_nextar_v2/model")
TF_MODEL = TFModel(ASSETS_PATH)

@app.route('/')
def hello():
    return py_file_location

@app.post('/predict')
@cross_origin()
def predict_image():
    req = request.get_json(force=True)
    image = _process_base64(req)
    return TF_MODEL.predict(image)

def _process_base64(json_data):
    image_data = json_data.get("image")
    image_data = re.sub(r"^\data:image/.+;base64,", "", image_data)
    image_base64 = bytarray(image_data, "utf8")
    image = base64.decodebytes(image_base64)
    return Image.open(io.BytesIO(image))

# Start the Flask server in a new thread
threading.Thread(target=app.run, kwargs={"use_reloader": False}).start()
# if __name__ == "__main__":
#     app.run(host='0.0.0.0', port=5000, debug=True)

```

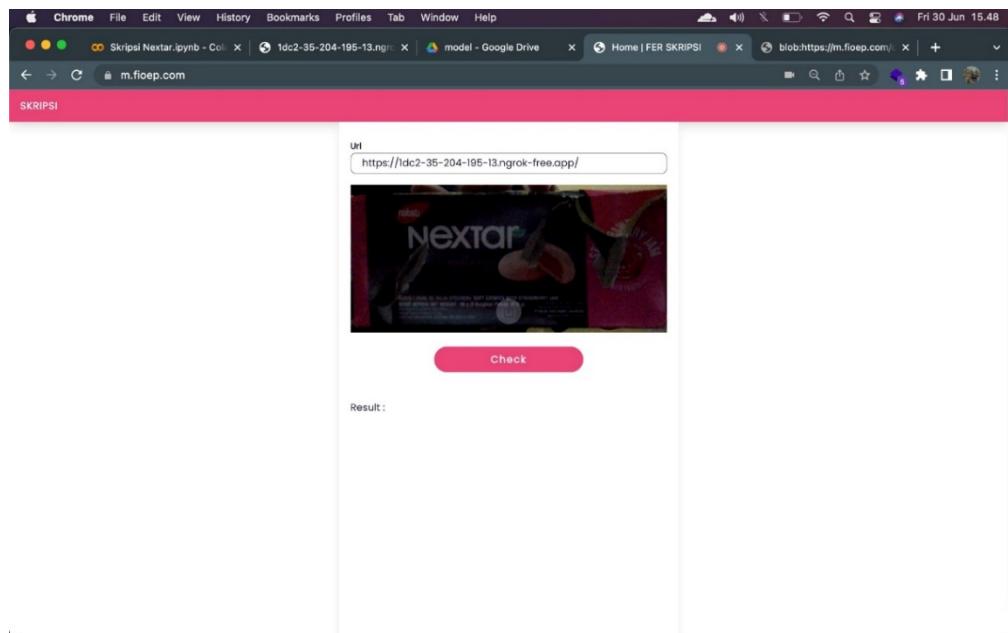
Gambar 4.2. Kode Pada Google Collab

Jalankan Kode diatas maka akan muncul URL publik yang bisa diakses. Setelah itu copas url tersebut untuk dimasukkan ke halaman utama *website frontend*.

4.1.5 Penyiapan *Web Frontend*

Halaman ini berguna untuk mengambil gambar Nextar dari pengguna untuk diupload ke *backend python* untuk dilakukan cek data apakah gambar tersebut merupakan gambar Nextar yang rusak atau bagus. Untuk menyiapkan website ini supaya bisa diakses oleh publik bisa menggunakan Vercel. Vercel adalah *website* yang menyediakan fasilitas

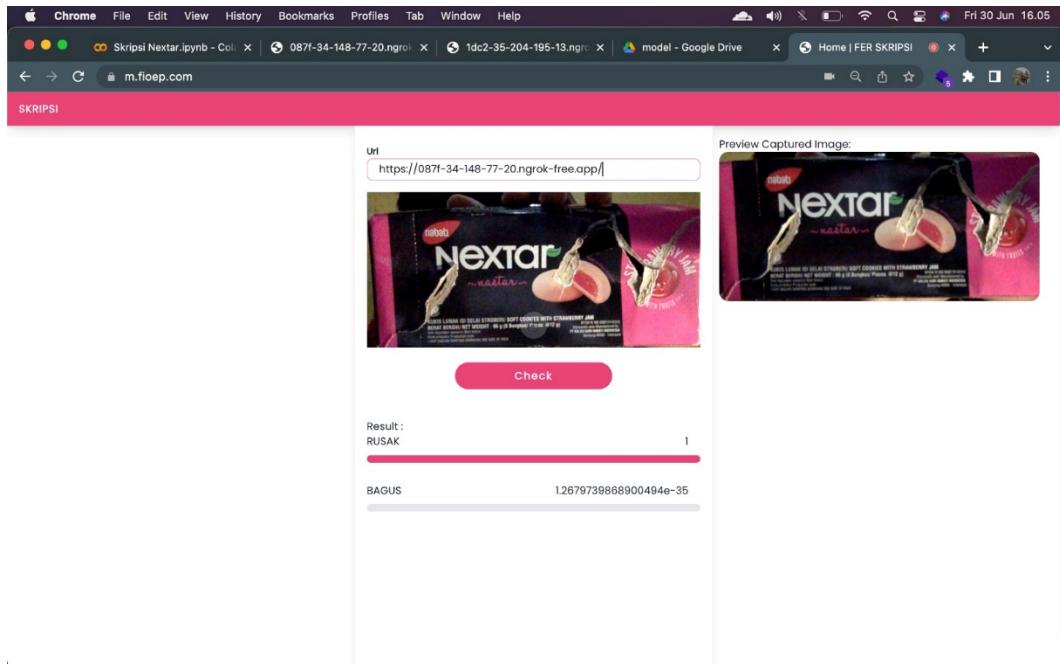
untuk melakukan *build Framework Javascript* secara gratis supaya dapat diakses oleh publik. Setelah melakukan *build* dan mendapatkan url publik dari Vercel segera akses url tersebut. Setelah muncul tampilan *web*nya berikan akses video karena *website* ini memerlukan akses untuk mengambil video atau gambar untuk melakukan validasi gambar secara *realtime*. Dan masukkan url yang didapatkan dari *Backend* pada menu url. Untuk pengiriman gambar yang dilakukan pada *frontend* ke *backend* menggunakan gambar *base-64* dari hasil capture camera yang berformatkan JPEG



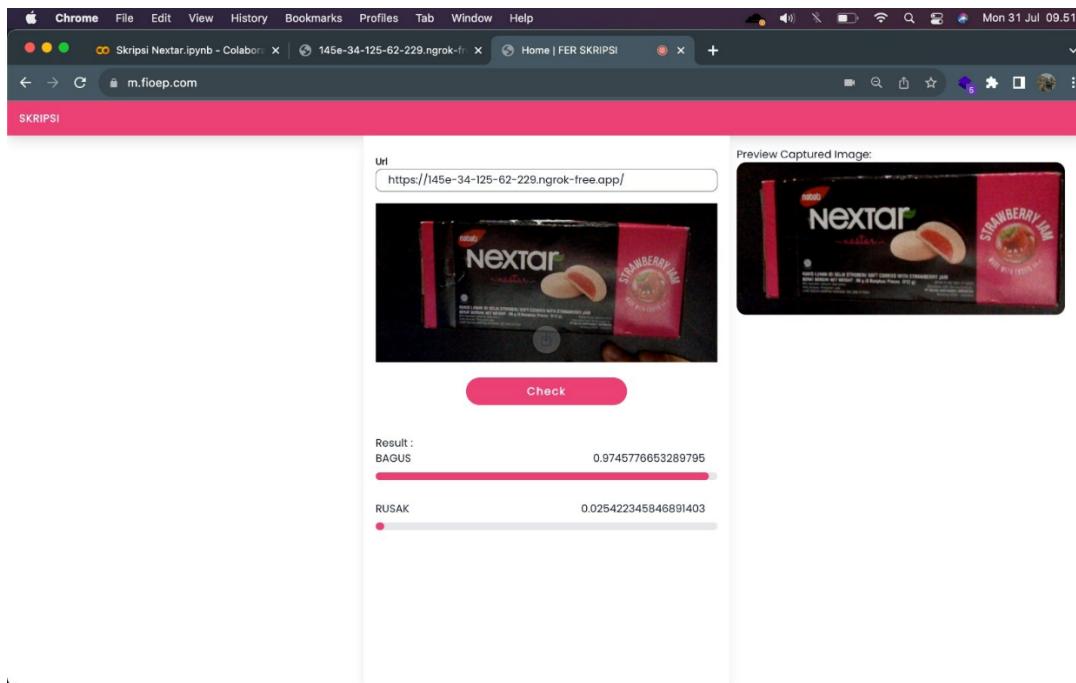
Gambar 4.3. Halaman Utama *Website*

4.1.6 Validasi

Proses ini dilakukan untuk mendapatkan label antara bagus dan rusak dari inputan gambar yang dilakukan pengguna menggunakan kamera video secara *realtime*. Untuk mengambil dan mengirim gambar bisa menggunakan tombol enter secara otomatis akan mengambil gambar dari video lalu dikirim ke *backend* untuk divalidasi dan mendapatkan label.



Gambar 4.4. Hasil Validasi dari *website* apabila rusak



Gambar 4.5. Hasil Validasi dari *website* apabila bagus

4.2 Pembahasan

Pembahasan perhitungan dan analisis metrik evaluasi untuk mengevaluasi performa model yang telah dilatih dalam bentuk scenario menggunakan metrik evaluasi sebagai berikut.

4.1. Pembahasan

Pembahasan perhitungan dan analisis metrik evaluasi untuk mengevaluasi performa model yang telah dilatih sebagai berikut.

4.1.1. Evaluasi Metrik

Dalam evaluasi model, kami menggunakan beberapa metrik evaluasi yang umum digunakan dalam tugas klasifikasi, yaitu Akurasi, Presisi, *Recall*, dan *F1-Score*.

4.1.1.1. Akurasi

Akurasi mengukur sejauh mana model dapat mengklasifikasikan dengan benar seluruh data dalam dataset. Kami menghitung akurasi menggunakan rumus: Akurasi = $\frac{(TP + TN)}{(TP + TN + FP + FN)}$.

TP = *True Postif* adalah label bagus dengan hasil yang benar

TN = *True Negatif* adalah label rusak dengan hasil yang benar

FP = *False Postif* adalah label bagus dengan hasil yang salah

FN = *False Negatif* adalah label rusak dengan hasil yang salah

4.1.1.2. Presisi

Presisi mengukur sejauh mana prediksi positif yang dibuat oleh model adalah benar. Kami menghitung presisi menggunakan rumus:

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

4.1.1.3. Recall

Recall mengukur sejauh mana model mampu mendeteksi dengan benar data positif. Kami menghitung recall menggunakan rumus: Recall =

$$\frac{TP}{(TP + FN)}$$

4.1.1.4. F1-Score

F1-Score merupakan nilai rata-rata harmonik antara presisi dan recall, memberikan ukuran keseluruhan performa model. Kami menghitung *F1-Score* menggunakan rumus: $F1\text{-Score} = \frac{2 * (\text{Presisi} * \text{Recall})}{(\text{Presisi} + \text{Recall})}$

4.2.1 Perhitungan

Berikut hasil perhitungan metrik evaluasi dari hasil dari training model yang dilakukan dengan beberapa jumlah data yang berbeda. Sebagai berikut:

Jumlah								
Data	TN	FN	TP	FP	Akurasi	Presisi	Recall	F1-Score
30	13	2	13	2	0,933333	0,866667	0,866667	0,866667
80	30	3	40	7	0,9125	0,851064	0,930233	0,888889
134	69	2	60	1	0,992424	0,983607	0,967742	0,97561
215	69	2	144	1	0,99537	0,993103	0,986301	0,989691
411	206	2	202	3	0,992736	0,985366	0,990196	0,987775

Tabel. 4.2. Hasil perhitungan matrik pada pelatihan model

JD : Jumlah data

P : Percobaan yang dilakukan

TN : True Negative (Label rusak dengan prediksi benar)

FN : False Negative (Label rusak dengan prediksi salah)

TP : True Positive (Label bagus dengan prediksi benar)

FP : False Posittive (Label bagus dengan prediksi salah)

Analisis hasil perhitungan metrik evaluasi menunjukkan bahwa model yang telah dilatih dengan menggunakan jumlah dataset yang lebih banyak mampu mengklasifikasikan kemasan Nextar yang rusak dengan tingkat akurasi, *presisi*, *recall* dan *f1-score* yang tinggi menunjukkan bahwa dari semua data yang diprediksi mampu melakukan validasi inputan data yang lebih efektif. *F1-Score* sebesar 98.77% menunjukkan

keseimbangan antara *presisi* dan *recall* dalam memprediksi kemasan biskuit Nextar yang rusak. Selain itu dilakukan juga hasil uji coba pada aplikasi yang dilakukan selama beberapa kali dengan jumlah data yang berbeda dan dilakukan perhitungan metrik evaluasi terhadap hasil uji coba yang dilakukan sebagai berikut :

JD	P	TN	FN	TP	FP	Akurasi	Presisi	Recall	F1-Score
b30	100	15	35	16	34	0,66	0,32	0,313725	0,316832
d34	100	40	10	38	12	0,88	0,76	0,791667	0,77551
215	100	32	18	42	8	0,92	0,84	0,7	0,763636
411	100	45	5	46	4	0,96	0,92	0,901961	0,910891

3. Hasil perhitungan matrik saat uji coba pada aplikasi

Analisis hasil perhitungan metrik evaluasi menunjukkan bahwa model yang telah dilatih dengan menggunakan jumlah dataset yang lebih sedikit bisa saja mendapatkan nilai akurasi yang lebih bagus dari nilai akurasi dengan dataset yang lebih banyak. Sebagai contoh adalah jumlah dataset 215 dengan jumlah dataset 411 memiliki tingkat akurasi yang lebih tinggi pada saat melakukan analisis hasil perhitungan metrik evaluasi pelatihan model dengan nilai akurasi 0,99537 yang lebih besar dari akurasi dataset 411 yang bernilai 0,992736. Namun pada saat melakukan analisis hasil perhitungan metrik evaluasi pada saat melakukan uji coba pada aplikasi *web* hasil akurasi menunjukkan nilai yang lebih bagus pada dataset yang lebih banyak seperti pada jumlah dataset 215 dan 411. Akurasi pada nilai tersebut adalah 0,92 dan 0,96.

BAB V

KESIMPULAN

5.1. Kesimpulan

Penelitian ini menggunakan teknik *Deep Learning* dengan model *Tensorflow* dan CNN sebagai metode penelitian menggunakan bantuan aplikasi Lobe dalam pengembangan pendekripsi kemasan biskuit nextar yang rusak. Pada tahapannya peneliti melakukan pengumpulan pembuatan data, pengujian data, penerapan data, dan analisis evaluasi metrik.

Hasil analisis yang didapatkan dari perhitungan metrik evaluasi menunjukkan bahwa model yang telah dilatih mampu mengklasifikasikan kemasan Nextar yang rusak dengan tingkat akurasi yang tinggi, walaupun dengan data yang sedikit seperti dengan dataset 215, dan akurasi sebesar 0,99537 sedangkan dataset 411 dengan nilai akurasi 0,992736. Namun setelah dilakukan analisis evaluasi metrik ulang menggunakan data uji coba yang dilakukan pada aplikasi *Web* menunjukkan bahwa dataset dengan data lebih banyak memiliki hasil akurasi yang lebih baik daripada dataset lebih kecil.

Training model yang dilakukan menggunakan Lobe mampu melakukan prediksi yang bagus dengan mendapatkan akurasi 0,992736 pada jumlah data 411. Dalam melakukan proses training pada aplikasi Lobe memerlukan koneksi internet dan sistem komputer yang cukup untuk melaksanakan pemrosesan pelatihan model. Dengan interface yang simpel dan mudah digunakan membuat pengguna awam cukup merasakan kenyamanan dalam menggunakannya untuk membuat aplikasi yang membutuhkan deep learning atau sejenisnya. Namun hal tersebut berbanding terbalik apabila digunakan oleh pengguna yang menginginkan pelatihan model yang lebih *expert* untuk diteliti atau dikembangkan aplikasi Lobe ini sangat tidak cocok. Penggunaan yang simpel dan *interface* yang mudah dimengerti adalah nilai unggul dalam aplikasi ini.

Dalam pengembangan aplikasi web yang dikembangkan ini menggunakan platform website karena adanya kemudahan akses pada *smartphone* maupun komputer. Sistem kerja dari aplikasi yang dikembangkan yaitu aplikasi melakukan pengambilan gambar dari kemasan produk, setelah itu gambar akan dikirim ke *backend* untuk mendapatkan label dari kemasan bagus atau kemasan rusak. Dan untuk pengembangan lebih lanjut menggunakan sistem yang lebih komplek sangat memungkinkan dilakukan karena dibangun terpisah antara *backend* dan *frontend*

5.2. Saran

Adapun untuk saran dari hasil penelitian ini, yaitu:

- a. Akurasi dari pelatihan model dari *tool* Lobe mampu memberikan hasil yang bagus dengan jumlah dataset yang lebih banyak.
- b. Pelatihan yang dilakukan dalam penelitian ini menggunakan *tool* Lobe untuk membuat model *tensorflow* tetapi dengan *interface* yang mudah digunakan.
- c. Disarankan dalam pengambilan dataset sesuai dengan lingkungan yang digunakan dalam penerapan aplikasinya sebab kecerahan ruangan sangat mempengaruhi hasil dari prediksi aplikasi
- d. Penerapan aplikasi ini disatukan dengan alat – alat yang ada di lingkungan pabrik sehingga sistem menjadi lebih efisien.
- e. Pengintegrasian aplikasi ini ke sistem informasi yang sudah berjalan di pabrik supaya dapat dianalisa data yang didapat

DAFTAR PUSTAKA

- Fiddiyansyah, R., Ana Wati, S. F., Fitri, A. S., Zidane, F. H., & Kuslaila, N. R. (2023). ANALISIS DAN PERANCANGAN SISTEM PRESENSI MAHASISWA BERBASIS TEKNOLOGI PENGENALAN WAJAH DI FAKULTAS ILMU KOMPUTER UPN VETERAN JAWA TIMUR. *Jurnal Informatika dan Teknik Elektro Terapan*, 11(1). <https://doi.org/10.23960/jitet.v11i1.2868>
- Harani, N. H., Prianto, C., & Hasanah, M. (2019). *Deteksi Objek Dan Pengenalan Karakter Plat Nomor Kendaraan Indonesia Menggunakan Metode Convolutional Neural Network (CNN) Berbasis Python*. Vol.11 No.3.
- Harika, M., Ramdania, D. R., Hidayat, R. S., Oktarini, S., & Feirizal, F. (2022). Penerapan Klasifikasi Untuk Kelayakan Hasil Produksi Jam Tangan dengan Menggunakan Algoritma K-Nearest Neighbor. *JURIKOM (Jurnal Riset Komputer)*, 9(6), 1850. <https://doi.org/10.30865/jurikom.v9i6.5216>
- Hikmatia A.E, N., & Ihsan Zul, M. (2021). Aplikasi Penerjemah Bahasa Isyarat Indonesia menjadi Suara berbasis Android menggunakan Tensorflow. *Jurnal Komputer Terapan*, Vol. 7 No. 1 (2021), 74–83. <https://doi.org/10.35143/jkt.v7i1.4629>
- Ihsan, O. M., Verina, W., Dewi, R., & Tanjung, D. H. (2023). *Perbandingan Konvensional Method dengan Fast Fourire Transform Method pada Efisiensi Citra Digital*. 7(2).
- Lesmana, A. M., Fadhillah, R. P., & Rozikin, C. (2022). Identifikasi Penyakit pada Citra Daun Kentang Menggunakan Convolutional Neural Network (CNN). *Jurnal Sains dan Informatika*, 8(1), 21–30. <https://doi.org/10.34128/jsi.v8i1.377>
- Listyalina, L., Yudianingsih, Y., Soedjono, A. W., Utari, E. L., & Dharmawan, D. A. (2022). Deep-RIC: Plastic Waste Classification using Deep Learning and Resin Identification Codes (RIC). *Telematika*, 19(2), 215. <https://doi.org/10.31315/telematika.v19i2.7419>

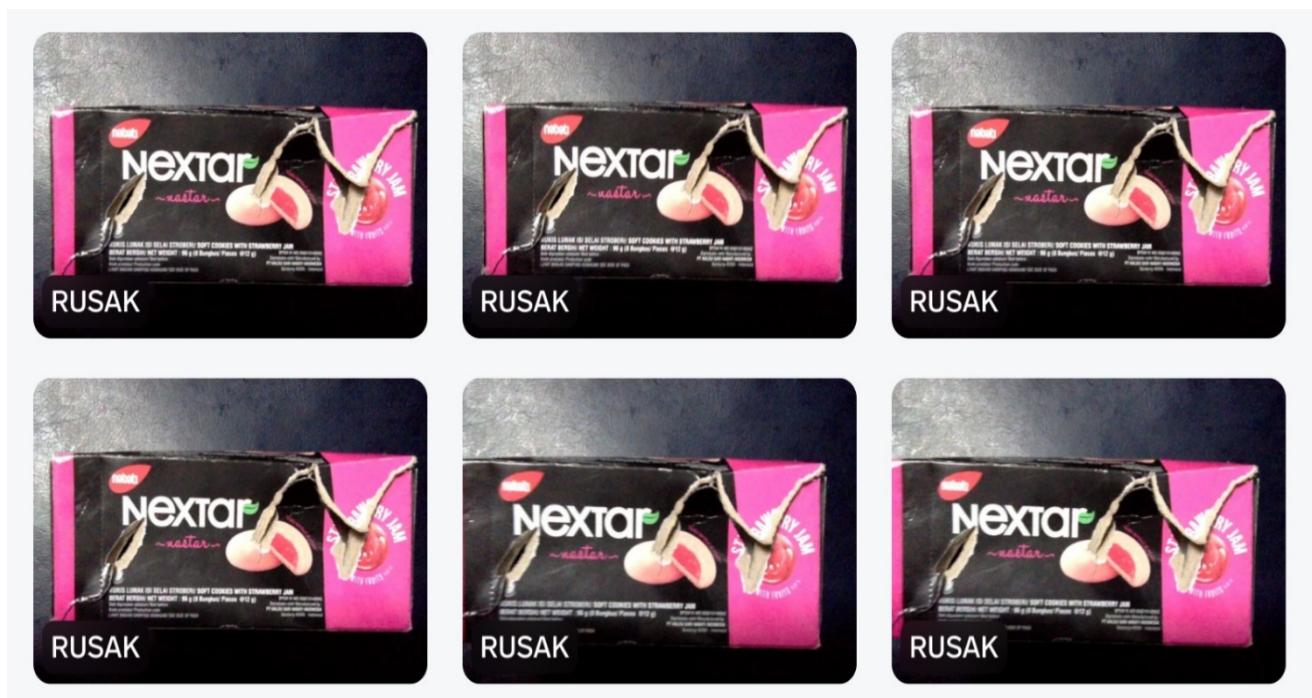
- Priyanti, E. (2021). Deteksi Bakteri Pada Produk Makanan Kemasan Menggunakan Algoritma Naïve Bayes. *IMTechno: Journal of Industrial Management and Technology*, 2(1).
- Raup, A., Ridwan, W., Khoeriyah, Y., Supiana, S., & Zaqiah, Q. Y. (2022). Deep Learning dan Penerapannya dalam Pembelajaran. *JIIP - Jurnal Ilmiah Ilmu Pendidikan*, 5(9), 3258–3267. <https://doi.org/10.54371/jiip.v5i9.805>
- Ridha, N. (2017). *PROSES PENELITIAN, MASALAH, VARIABEL DAN PARADIGMA PENELITIAN*. 14(1).
- Rorong, J. A., & Wilar, W. F. (2020). *KERACUNAN MAKANAN OLEH MIKROBA*. 2(2).
- Setyaningsih, E. R., & Edy, M. S. (2022). YOLOv4 dan Mask R-CNN Untuk Deteksi Kerusakan Pada Karung Komoditi. *TEKNIKA*, 11(1), 45–52. <https://doi.org/10.34148/teknika.v11i1.419>
- Sugiono. (2019). *METODE PENELITIAN PENDIDIKAN*. ALFABETA.
- TiaraSari, A., & Haryatmi, E. (2021). Penerapan Convolutional Neural Network Deep Learning dalam Pendekripsi Citra Biji Jagung Kering. *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, 5(2), 265–271. <https://doi.org/10.29207/resti.v5i2.3040>
- Trisiawan, I. K., & Yuliza, Y. (2022). Penerapan Multi-Label Image Classification Menggunakan Metode Convolutional Neural Network (CNN) Untuk Sortir Botol Minuman. *Jurnal Teknologi Elektro*, 13(1), 48. <https://doi.org/10.22441/jte.2022.v13i1.009>
- Valentina, R., Rostianingsih, S., & Tjondrowiguno, A. N. (2020). *Pengenalan Gambar Botol Plastik dan Kaleng Minuman Menggunakan Metode Convolutional Neural Network*.
- Vu, T.-T.-H., Pham, D.-L., & Chang, T.-W. (2023). A YOLO-based Real-time Packaging Defect Detection System. *Procedia Computer Science*, 217, 886–894. <https://doi.org/10.1016/j.procs.2022.12.285>

LAMPIRAN

Lampiran 1. Gambar dataset kemasan bagus



Lampiran 2. Gambar kemasan rusak



Lampiran 3. Kode *frontend* utama

indexjs

```
<script setup>
import axios from "axios";
import { onMounted, ref, watch } from "vue";
import Camera from "simple-vue-camera";

const size = ref({
  width: 430,
  height: 200,
});
const log_ = ref("");
const loading = ref(false);
const preview = ref();
const result = ref([]);
const camera = ref();
// Use camera reference to call functions
const snapshot = async () => {
  const blob = await camera.value?.snapshot();

  // To show the screenshot with an image tag, create a url
  const url = URL.createObjectURL(blob);
  preview.value = url;
  encodeBase(blob);
};
```

indexjs

```
const forme = ref({
  url: "http://127.0.0.1:5000/",
  image: "",
});
function loge(e) {
  log_.value = log_.value + "<br />" + e;
}

watch(
  () => forme.value.url,
  (e) => {
    localStorage.setItem("url", e);
  }
);
onMounted(() => {
  forme.value.url = localStorage.getItem("url");
  window.addEventListener("keydown", (e) => {
    if (e.key == "Enter") {
      snapshot();
    }
  });
});
async function postImg() {
  // log("Loading.....");
  log_.value = "";
  loading.value = true;
  result.value = [];
  try {
    const { data } = await axios.post(`${forme.value.url}predict`, {
      image: forme.value.image,
    });
    // log("Success.....");
    // let tmp0 = data.predictions[0];
    // let tmp1 = data.predictions[1];
    // if (tmp0.confidence > tmp1.confidence) {
    //   log(` ${tmp0.label} with Confidence ${tmp0.confidence}`);
    // } else {
    //   log(` ${tmp1.label} with Confidence ${tmp1.confidence}`);
    // }
    result.value = data.predictions;
  } catch (e) {
    console.log(e);
    log("Error Response.....");
  } finally {
    loading.value = false;
  }
}
function encodeBase(file) {
  preview.value = URL.createObjectURL(file);
  let reader = new FileReader();
  reader.onloadend = function () {
    forme.value.image = reader.result;
    postImg();
  };
  reader.readAsDataURL(file);
}
function encodeImageFileAsURL(e) {
  let file = e.target.files[0];
  // console.log(e);
  // return;

  preview.value = URL.createObjectURL(file);
  let reader = new FileReader();
  reader.onloadend = function () {
    // console.log("RESULT", reader.result);
    forme.value.image = reader.result;
  };
  reader.readAsDataURL(file);
}
</script>
```