

**INOVASI HYBRID NEURAL NETWORK ALGORITMA SINGLE SHOT MULTIBOX
DETECTOR (SSD) DAN RESNET -18 IDENTIFIKASI DAN KLASIFIKASI SAMPAH
PESISIR (STUDI KASUS PANTAI OLO MEDAN)**

SKRIPSI

Farrel Dwi Prayogo

201401119



**PROGRAM STUDI S-1 ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA MEDAN**

2024

**INOVASI HYBRID NEURAL NETWORK ALGORITMA SINGLE SHOT MULTIBOX
DETECTOR (SSD) DAN RESNET -18 IDENTIFIKASI DAN KLASIFIKASI SAMPAH
PESISIR (STUDI KASUS PANTAI OLO MEDAN)**

SKRIPSI

Diajukan untuk melengkapi tugas dan memenuhi syarat memperoleh ijazah
Sarjana Komputer

Farrel Dwi Prayogo
201401119



PROGRAM STUDI S-1 ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA
MEDAN
2024

PERSETUJUAN

Judul : INOVASI HYBIRD NEURAL NETWORK
 ALGORITMA SINGLE SHOT MULTIBOX
 DETECTOR (SSD) DAN RESNET -18
 IDENTIFIKASI DAN KLASIFIKASI SAMPAH
 PESISIR (STUDI KASUS PANTAI OLO MEDAN

Kategori : SKRIPSI

Nama : Farrel Dwi Prayogo

Nomor Induk Mahasiswa : 201401119

Program Studi : SARJANA (S-1) ILMU KOMPUTER

Fakultas : ILMU KOMPUTER DAN
 TEKNOLOGI INFORMASI
 UNIVERSITAS SUMATERA
 UTARA

Komisi Pembimbing :

Dosen Pembimbing I

Dosen Pembimbing II

Dr. T. Henny Febriana
 Harumy, S.Kom., M.Kom
 NIP. 197812212014042001

Prof. Dr. Syahril Efendi S.Si., MIT.
 NIP. 196711101996021001

Diketahui/disetujui oleh
 Program Studi S1 Ilmu Komputer
 Ketua,

Dr. Amalia S.T M.T
 NIP. 197812212014042001

PERNYATAAN

**INOVASI HYBRID NEURAL NETWORK ALGORITMA SINGLE SHOT MULTIBOX
DETECTOR (SSD) DAN RESNET -18 IDENTIFIKASI DAN KLASIFIKASI SAMPAH PESISIR
(STUDI KASUS PANTAI OLO MEDAN)**

SKRIPSI

Saya mengakui bahwa skripsi ini adalah hasil karya saya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing disebutkan sumbernya.

Medan, 18 Juni 2024

Farrel Dwi Prayogo
201401119

UCAPAN TERIMA KASIH

Puji Alhamdulillah, penulis merasa sangat bersyukur dan berterima kasih kepada Allah SWT atas rahmat dan kekuasaan-Nya yang telah memberikan kemudahan dan kesempatan bagi penulis untuk menyelesaikan skripsi ini sebagai persyaratan untuk memperoleh gelar Sarjana Komputer dari Program Studi S-1 Ilmu Komputer Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.

Penulis ingin mengekspresikan rasa hormat dan terima kasih yang tulus kepada semua pihak yang telah memberikan bantuan dan dukungan dalam proses penulisan dan penyelesaian skripsi ini. Terima kasih yang sebesar-besarnya untuk mereka yang telah memberikan doa, bimbingan, kerjasama, dukungan, serta kata-kata yang memberikan ketenangan dalam perjalanan penulisan skripsi ini. Penulis mengapresiasi semua bantuan dan kontribusi yang telah diberikan dari berbagai pihak. Penulis mengucapkan terima kasih kepada:

1. Prof. Dr. Muryanto Amin S.Sos., M.Si. sebagai Rektor Universitas Sumatera Utara
2. Dr. Maya Silvi Lydia B.Sc., M.Sc. sebagai Dekan Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
3. Dr. Amalia S.T., M.T. sebagai Ketua Program Studi S- 1 Ilmu Komputer Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
4. Ibu Dr T. Henny Febriana Harumy S.Kom., M.Kom sebagai dosen pembimbing I yang telah memberikan banyak masukan, kritik dan saran terhadap penulis sehingga penulis dapat menyelesaikan skripsi ini.
5. Bapak Prof. Dr. Syahril Efendi S.Si., MIT. selaku dosen pembimbing II yang telah banyak memberikan ilmu dan masukan terhadap skripsi yang dikerjakan oleh penulis.
6. Kedua orangtua penulis, Ir.Suprayogi dan Dewi Murti, yang senantiasa menyayangi, mendoakan, dan mendukung penulis dalam tiap aspek kehidupan yang penulis lalui.

7. Teman-teman Mujahidin sekaligus seluruh Stambuk 2020 yang mengenal saya dan membantu saya.
8. Teman-Teman seperjuangan : Haryanda Fidi , Ayu Prasinta , Ainun Mardiah , Rani Syafitri , Syabrina Ramadhani Kamal , Tridinda

Semoga hasil penelitian ini dapat memberikan kontribusi dan manfaat yang nyata bagi perkembangan ilmu pengetahuan dan teknologi. Penulis menyadari bahwa penelitian ini memiliki keterbatasan dan ruang untuk pengembangan lebih lanjut. Oleh karena itu, Penulis berharap penelitian ini dapat menjadi dasar untuk penelitian lanjutan dan menginspirasi penelitian lainnya.

Akhir kata, Terima kasih kepada saya sendiri yang sudah berjuang dan tidak menyerah sampai sekarang sehingga dapat menyelesaikan skripsi ini. semoga skripsi ini bermanfaat bagi pembaca dan masyarakat luas. Penulis berharap agar hasil penelitian ini dapat memberikan sumbangsih positif dalam memahami sentimen pengguna dalam konteks digital. Terima kasih

Medan, 15 Mei 2024

Penulis

ABSTRAK

Sampah pesisir merupakan permasalahan lingkungan yang serius di Indonesia, termasuk di Pantai Olo Medan. Pengelolaan sampah pesisir yang tidak optimal dapat menyebabkan pencemaran laut, kerusakan ekosistem, dan gangguan kesehatan masyarakat. Oleh karena itu, diperlukan metode yang efektif untuk identifikasi dan klasifikasi sampah pesisir. Penelitian ini mengusulkan inovasi hybrid neural network algoritma *Single Shot Multibox Detector (SSD)* dan *ResNet-18* untuk identifikasi dan klasifikasi sampah pesisir. Algoritma *SSD* digunakan untuk mendeteksi lokasi sampah pesisir dalam gambar, sedangkan *ResNet-18* digunakan untuk mengklasifikasikan jenis sampah pesisir. Model hybrid ini dilatih dengan menggunakan dataset gambar sampah pesisir Pantai Olo Medan. Hasil penelitian menunjukkan bahwa model hybrid *SSD* dan *ResNet-18* memiliki akurasi yang tinggi dalam identifikasi dan klasifikasi sampah pesisir. Akurasi deteksi mencapai 95%, sedangkan akurasi klasifikasi mencapai 90%. Model ini dapat mendeteksi dan mengklasifikasikan berbagai jenis sampah pesisir, seperti sampah plastik, sampah organik, dan sampah logam. Inovasi hybrid neural network algoritma *SSD* dan *ResNet-18* memiliki potensi untuk menjadi metode yang efektif untuk identifikasi dan klasifikasi sampah pesisir. Model ini dapat digunakan untuk membantu pengelolaan sampah pesisir yang lebih optimal.

Kata Kunci: *Sampah Pesisir, Convolutional Neural Network (CNN), , Single Shot MultiboxDetector (SSD), ResNet-18, Pantai Olo Medan*

ABSTRACT

Coastal waste is a serious environmental problem in Indonesia, including at Olo Beach Medan. Inefficient coastal waste management can lead to marine pollution, ecosystem damage, and public health problems. Therefore, an effective method for identifying and classifying coastal waste is needed. This study proposes a hybrid neural network innovation using the Single Shot Multibox Detector (SSD) and *ResNet-18* algorithms for coastal waste identification and classification. The SSD algorithm is used to detect the location of coastal waste in images, while *ResNet-18* is used to classify the type of coastal waste. This hybrid model is trained using a dataset of coastal waste images from Olo Beach Medan. The results show that the hybrid SSD and *ResNet-18* model has high accuracy in identifying and classifying coastal waste. Detection accuracy reaches 95%, while classification accuracy reaches 90%. The model can detect and classify various types of coastal waste, such as plastic waste, organic waste, and wood waste. The hybrid neural network innovation using the SSD and *ResNet-18* algorithms has the potential to be an effective method for coastal waste identification and classification. This model can be used to assist in more optimal coastal waste management at Olo Beach Medan and other coastal areas.

Keywords: *Coastal Waste, Deep Learning, Convolutional Neural Network (CNN), Identification, Classification, Single Shot Multibox Detector (SSD), ResNet-18, Olo Beach Medan.*

DAFTAR ISI

SKRIPSI	II
PERSETUJUAN	III
PERNYATAAN	IV
UCAPAN TERIMA KASIH	V
ABSTRAK	VII
ABSTRACT	VIII
DAFTAR ISI.....	IX
DAFTAR TABEL.....	XI
DAFTAR GAMBAR.....	XII
DAFTAR LAMPIRAN.....	XIV
BAB 1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Batasan Masalah.....	2
1.4 Tujuan Penelitian.....	3
1.5 Manfaat Penelitian.....	3
1.6 Metodologi Penelitian	3
1.7 Sistematika Penyusunan	5
BAB 2 LANDASAN TEORI	7
2.1 Pantai Olo.....	7
2.2 Sampah.....	7
2.3 Machine Learning	8
2.4 Deep Learning.....	10
2.5 <i>Artificial Neural Network</i>	11
2.6 Convolutional Neural Network.....	13
2.7 Tensor Flow	13
2.8 Single Shot Multibox Detector (SSD)	14
2.9 Resnet -18	18
2.10 Roboflow.....	21
2.11 Penelitian yang Relevan	22
BAB 3.....	24
ANALISIS DAN PERANCANGAN	24

3.1	Arsitektur Umum.....	24
3.2	Labelisasi Dataset.....	25
3.3	Pengolahan.....	27
3.4	Pre-processing Dataset	28
3.5	Implementasi <i>SSD & Resnet-18</i>	30
3.6	Diagram Alur Sistem	31
3.7	Proses Klasifikasi dan Identifikasi Citra dan perancangan Antarmuka...	32
BAB 4 IMPLEMENTASI DAN PENGUJIAN		35
4.1	Implementasi Sistem	35
4.1.1.	Spesifikasi Perangkat Keras	35
4.1.2.	Spesifikasi Perangkat Lunak	35
4.2	Implementasi Pengambilan Dataset	36
4.3	Labelisasi Dataset.....	37
4.4	Preprocessing Dataset.....	39
4.4.1.	Data Resizing	40
4.4.2.	Normalisasi	40
4.5	Split Dataset	41
4.6	Implementasi Resnet-18	41
4.7	Implementasi Singel Shot Multibox Detector (SSD)	47
4.8	Implementasi Tahapan Antar Muka.....	51
4.8.1.	Halaman Dashboard (Landing Page)	52
4.8.2.	Halaman Upload (Uji coba data)	52
4.9	Pengujian Sistem.....	53
4.10	Evaluasi.....	59
BAB 5 KESIMPULAN DAN SARAN		62
5.1	Kesimpulan.....	62
5.2	Saran.....	63
DAFTAR PUSTAKA		64
LAMPIRAN.....		66

DAFTAR TABEL

Table 1. 1	Pengujian Sistem Implementasi Website	53
Table 1. 2	Perhitungan Nilai Akurasi Algoritma ResNet18	59
Table 1. 3	Menghitung Nilai akurasi Algoritma SSD	60

DAFTAR GAMBAR

Gambar 2. 1 Hubungan antara Deep Learning, Machine Learning dan Kecerdasan Buatan (Chollet, 2018)	11
Gambar 2. 2 Multilayered perceptron (Fatehnia & Amirinia, 2018).....	12
Gambar 2. 3 CNN Arsitektur VGG-16 (Ali Rohan., et al., 2020)	15
Gambar 2. 4 Blok Residual Learning Resnet (Nofal Anam , 2022)	18
Gambar 2. 5 Arsitektur ResNet-18,(H. Zayd et al., 2022).....	19
Gambar 2. 6 Roboflow Image Profile (https://roboflow.com)	21
Gambar 3. 1 Arsitektur Sistem	24
Gambar 3. 2 Sample Gambar.....	27
Gambar 3. 3 Diagram Image Processing	28
Gambar 3. 4 Diagram Activity	31
Gambar 3. 5 Diagram Sequence	31
Gambar 3. 6 Dashboard Pages	33
Gambar 3. 7 Gambar Halaman Hasil Uji.....	34
Gambar 4.1 Jenis Sampah Pesisir	36
Gambar 4.2 Proses Scrapping dataset kaggle	36
Gambar 4.3 Dataset Gdrive kategori Plastic.....	37
Gambar 4.4 Labelisasi Dataset Resnet-18	37
Gambar 4.5 Organik	38
Gambar 4.6 B3.....	38
Gambar 4.7 Anorganik	38
Gambar 4.8 Labelisasi Class.....	39
Gambar 4.9 Data resizing 128 x 128 Resnet.....	40
Gambar 4.10 Transformasi Data.....	40
Gambar 4.11 Akurasi Diperoleh pada 5 kali Loop Pelatihan dengan 10 Epoch	42
Gambar 4.12 Grafik Loss setiap Epoch	43
Gambar 4.13 Nilai Accuracy setiap Epoch.....	43
Gambar 4.14 Proses Training dan Evaluasi Dataset.....	45
Gambar 4.15 Perbandingan learning rate dan batch size pada setiap epoch	45

Gambar 4 16 Klasifikasi Resnet-18	46
Gambar 4.17 Proses Bounding Box SSD	47
Gambar 4 18 Training Data pada dataset Roboflow	48
Gambar 4.19 Grafik Pelatihan Model	49
Gambar 4.20 Validasi Akurasi SSD	50
Gambar 4.21 Implementasi dua model Algoritma.....	51
Gambar 4.22 Implementasi Halaman Pertama	52
Gambar 4.23 Implementasi Halaman Hasil Uji.....	53
Gambar 4.24 Confusion Matrix ResNet-18	59
Gambar 5. 1 Confusion Matrix ResNet18	60

DAFTAR LAMPIRAN

Lampiran 1. Listing Program	A-1
-----------------------------------	-----

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Perkembangan Pantai Olo, yang terletak di Kecamatan Medan Belawan, menjadi saksi bisu terhadap permasalahan serius sampah pesisir. Peningkatan urbanisasi dan aktivitas manusia di sekitar pantai telah meningkatkan akumulasi sampah, melibatkan berbagai jenis materi seperti plastik, kayu, dan botol. Dampak negatif yang mungkin ditimbulkan terhadap ekosistem, kehidupan laut, dan kesehatan masyarakat setempat mendorong pencarian solusi inovatif. Dalam konteks ini, penerapan teknologi Neural Network, dengan fokus pada algoritma *Single Shot Multibox Detector (SSD)* dan *ResNet-18*, menandai langkah maju dalam upaya mengidentifikasi dan mengklasifikasikan sampah pesisir.

Penerapan teknologi kecerdasan buatan Machine Learning dapat membantu mengatasi tantangan ini dengan mengenali pola kompleks pada data gambar. Selain itu, integrasi dengan *Single Shot Multibox Detector (SSD)*, menghasilkan teknologi deteksi objek yang cepat dan akurat, dapat meningkatkan efisiensi identifikasi sampah dalam gambar atau video. Penelitian ini memberikan kontribusi pada literatur dengan mengusulkan inovasi hybrid algoritma yang menggabungkan kekuatan *SSD* dan *ResNet-18* untuk mencapai tingkat akurasi dan efisiensi yang lebih tinggi dalam identifikasi dan klasifikasi sampah pesisir. Penggunaan *SSD* dalam mendeteksi objek secara real-time, memberikan keunggulan dalam respons cepat terhadap perubahan kondisi pantai, yang kritis untuk pengelolaan sampah yang efektif.

Pada penelitian yang dilakukan oleh Octavia,Putranto (2022) yang berjudul “ Implementasi Deep Learning dengan *Convolutional Neural Network* untuk Klasifikasi gambar sampah organik dan Anorganik “ menyampaikan telah berhasil mengidentifikasi sampah baik anorganik dan organik dengan Tingkat akurasi 93% menggunakan Algoritma ANN sebagai acuan dengan menggunakan data training sebanyak 223 gambar dan data testing 55 gambar.begitu juga dengan Algoritma SSD Pernando,yudhi(2023) berjudul “Deteksi jenis sampah secara Realtimr menggunakan Metode “SINGEL SHOT MULTIBOX DETECTOR”

menyebutkan bahwa Tingkat akurasi yang diperoleh sebanyak 95% dengan lebih dari 200 data training dan 30 data testing.

Berdasarkan latar belakang yang telah penulis uraikan, maka dilakukan penelitian dengan judul “ *Inovasi Hybrid Neural Network Algoritma Single Shot Multibox Detector (SSD) dan Resnet-18 Identifikasi dan Klasifikasi Sampah pesisir* ”.

1.2 Rumusan Masalah

Berdasarkan Latar Belakang Pengelolaan sampah pesisir di Pantai Olo Medan masih menjadi tantangan. Metode identifikasi dan klasifikasi sampah pesisir yang ada saat ini belum optimal, sehingga diperlukan metode yang lebih efektif dan efisien. Penelitian ini bertujuan untuk mengembangkan model hybrid neural network yang menggabungkan algoritma *Single Shot Multibox Detector (SSD)* dan *ResNet-18* untuk identifikasi dan klasifikasi sampah pesisir di Pantai Olo Medan. Kinerja model akan dievaluasi dengan menggunakan dataset gambar sampah pesisir Pantai Olo Medan. Manfaat dan keterbatasan penerapan model ini dalam pengelolaan sampah pesisir akan dianalisis. Selanjutnya, model ini akan diimplementasikan dalam sistem untuk monitoring dan identifikasi sampah pesisir di Pantai Olo Medan. Penelitian ini diharapkan dapat memberikan solusi yang efektif untuk identifikasi dan klasifikasi sampah pesisir di Pantai Olo Medan, sehingga dapat membantu dalam pengelolaan sampah pesisir yang lebih optimal dan berkelanjutan..

1.3 Batasan Masalah

Dalam penelitian ini, peneliti memiliki batasan terhadap penyelesaian masalah.

Adapun batasan masalah dalam pelaksanaan penelitian ini adalah:

1. Bahasa pemrograman yang digunakan adalah Python dan Html implementasi website
2. Penelitian dilakukan di sekitaran wilayah lokasi Pantai Olo dan masih mencakup wilayah Belawan dan sekitar
3. Kelas untuk klasifikasi jenis sampah pada metode *SSD* yaitu Kaca, Plastik, Logam, Textile, Biologis, Styrofoam, Rokok
4. Kelas untuk klasifikasi kategori sampah pada *Resnet-18* yaitu Organik, Anorganik,

B3(berbahaya) dan untuk B3 jenis gambar hanya mencakup Rokok tidak ada jenis lain berdasarkan yang ditemukan di lapangan

5. Data set diambil Hybrid melalui survey langsung dan scrapping pada Kaggle dengan Total sebanyak 800 data.

1.4 Tujuan Penelitian

Tujuan penelitian ini adalah mengembangkan dan menerapkan inovasi Hybrid Algoritma *Single Shot Multibox Detector (SSD)* dan *Resnet-18* untuk identifikasi dan klasifikasi sampah pesisir di Pantai Olo Medan. Penelitian ini bertujuan untuk mengidentifikasi tantangan utama dalam pengelolaan sampah pesisir, merancang serta mengembangkan model Hybrid Algoritma *ResNet-18* dan *SSD*, dan mengevaluasi akurasi identifikasi sampah dibandingkan dengan metode tradisional.

1.5 Manfaat Penelitian

Metode Penelitian Penelitian ini diharapkan memberikan manfaat berupa peningkatan efisiensi pengelolaan sampah pesisir di Pantai Olo Medan melalui implementasi Hybrid Algoritma *SSD* dan *Resnet-18*. Manfaat lainnya melibatkan preservasi lingkungan pesisir, menjadi model referensi untuk lokasi pantai lain, kontribusi pada pengembangan teknologi AI dalam pengelolaan lingkungan, peningkatan kesadaran masyarakat terhadap isu lingkungan, dan membuka peluang penelitian lanjutan terkait pengembangan algoritma dan aplikasi teknologi AI dalam konservasi lingkungan.

1.6 Metodologi Penelitian

Metodologi penelitian yang digunakan adalah sebagai berikut:

1. Studi Pustaka

Dalam metode ini, peneliti mengumpulkan dan menganalisis bahan-bahan pustaka yang relevan, seperti jurnal, buku, artikel, atau makalah, untuk memperoleh pemahaman yang mendalam tentang topik penelitian yang ingin diteliti. Tujuan utama dari studi pustaka adalah untuk menyusun kerangka pemikiran yang kuat dan mendasari penelitian yang akan dilakukan, serta memperoleh pemahaman yang komprehensif tentang perkembangan penelitian terkini, temuan-temuan sebelumnya, dan

pendekatan yang digunakan oleh peneliti lain dalam bidang yang sama.

2. Identifikasi Masalah

Dalam metode ini, peneliti melakukan analisis menyeluruh terhadap lingkungan atau domain tertentu untuk mengidentifikasi masalah yang ada, celah pengetahuan yang perlu diisi, atau situasi yang membutuhkan pemecahan masalah. Tujuan utama dari metode ini adalah untuk menentukan tujuan penelitian yang jelas dan relevan, serta merumuskan pertanyaan penelitian yang spesifik dan terarah.

3. Analisis Sistem

Pada tahap ini, Sistem ini terdiri dari modul akuisisi data, pre-processing data, pelatihan model, identifikasi dan klasifikasi, dan visualisasi hasil. Arsitektur sistem menggunakan client-server dengan aliran data dari pengguna ke server dan kembali. Sistem ini membutuhkan perangkat keras dan perangkat lunak yang memadai, serta data gambar sampah pesisir yang cukup untuk melatih model

4. Perancangan dan Implementasi Sistem

Metode penelitian Perancangan dan Implementasi Sistem berfokus pada pengembangan solusi teknologi yang dapat diimplementasikan untuk memecahkan masalah yang relevan dengan topik penelitian. Hasil penelitian ini berupa sebuah program yang akan melakukan *training* terhadap ratusan data dan menghasilkan model yang cerdas.

5. Pengujian Sistem

Metode penelitian Pengujian Sistem adalah pendekatan yang digunakan untuk menguji dan mengevaluasi sistem yang telah dirancang dan diimplementasikan dalam suatu penelitian. Tujuan dari metode ini adalah untuk memverifikasi dan memvalidasi fungsionalitas, kualitas, dan performa sistem yang dikembangkan. Proses pengujian sistem melibatkan serangkaian langkah yang sistematis dan terencana. Pertama, peneliti akan merancang rencana pengujian yang mencakup tujuan pengujian, skenario pengujian, data uji, dan kriteria keberhasilan. Selanjutnya, peneliti akan melakukan pengujian fungsionalitas sistem untuk memastikan bahwa sistem dapat beroperasi sesuai dengan kebutuhan dan spesifikasi yang

telah ditetapkan.

6. Analisis Hasil

Metode penelitian Analisis Hasil adalah pendekatan yang digunakan untuk menganalisis data atau hasil yang telah dikumpulkan dalam penelitian. Tujuan dari metode ini adalah untuk menginterpretasikan data yang diperoleh dari penelitian, menarik kesimpulan, dan menjawab pertanyaan penelitian yang telah dirumuskan.

7. Dokumentasi

Dokumentasi adalah pendekatan yang menggunakan analisis dan penelaahan dokumen atau sumber data yang telah ada sebagai sumber informasi utama dalam penelitian. Metode ini melibatkan pengumpulan, evaluasi, dan interpretasi dokumen-dokumen yang relevan dengan topik penelitian untuk mendapatkan pemahaman yang lebih baik tentang fenomena yang diteliti.

1.7 Sistematika Penyusunan

Adapun sistematika dalam penelitian ini terdiri atas:

BAB 1: Pendahuluan

Bab ini merupakan tahap awal dalam sebuah penelitian yang menjelaskan latar belakang penelitian, permasalahan yang diteliti, tujuan penelitian, rumusan masalah, kegunaan penelitian, batasan penelitian, dan sistematika penulisan. Bab ini memberikan gambaran umum tentang topik penelitian dan alasan mengapa penelitian tersebut dilakukan.

BAB 2: Landasan Teori

Bab landasan teori berisi rangkuman literatur terkait yang relevan dengan topik penelitian. Peneliti melakukan pencarian dan kajian terhadap sumber-sumber informasi seperti jurnal ilmiah, buku, makalah, atau artikel terkait topik penelitian. Tujuan dari bab ini adalah untuk memperoleh pemahaman yang komprehensif tentang topik penelitian, menunjukkan keberadaan kekosongan pengetahuan, dan mengfokuskan pentingnya penelitian. Pada bab ini akan dijelaskan beberapa teori tentang permasalahan yang akan dibahas dalam penelitian ini yaitu Sampah Pesisir,

Deep Learning, Convolutional Neural Network (CNN), Single Shot Multibox Detector (SSD), ResNet-18

BAB 3: Analisis dan Perancangan

Penelitian ini bertujuan mengembangkan model hybrid neural network untuk identifikasi dan klasifikasi sampah pesisir di Pantai Olo Medan. Model ini menggabungkan algoritma *SSD* dan *ResNet-18* untuk mencapai akurasi tinggi dalam mendeteksi dan mengklasifikasikan berbagai jenis sampah. dengan tujuan membedakan jenis dan kategori sampah pada berbagai aspek lingkungan

BAB 4: Implementasi dan Pengujian

Bab Implementasi dan Pengujian adalah bagian penting dalam sebuah penelitian yang fokus pada penerapan dan evaluasi praktis dari sistem atau solusi yang telah dirancang. Dalam bab ini, peneliti menjelaskan langkah-langkah yang diambil untuk menerapkan solusi yang telah dirancang ke dalam lingkungan nyata atau simulasi.

BAB 5: Kesimpulan dan Saran

Bab Kesimpulan dan Saran merangkum temuan penelitian, menjawab pertanyaan penelitian, dan mengemukakan kesimpulan yang didapat dari hasil penelitian.

BAB 2

LANDASAN TEORI

2.1 Pantai Olo

Nama "Pantai Olo" diambil dari nama tokoh terkenal di Medan, Olo Panggabean. Olo Panggabean adalah figur yang cukup dikenal di Medan dan Sumatera Utara, terutama pada era 1980-an hingga awal 2000-an. Ia dikenal sebagai salah satu pemimpin kelompok pemuda dan memiliki pengaruh yang cukup besar dalam komunitas setempat. Pantai Olo Medan tidak dikelola oleh pemerintah atau instansi resmi tertentu, melainkan oleh komunitas lokal dan warga sekitar yang memanfaatkan kawasan tersebut untuk kegiatan wisata. Biasanya, pengelolaan tempat seperti ini melibatkan penduduk setempat yang membuka warung makanan dan minuman, menyediakan fasilitas sederhana seperti tempat duduk dan area parkir, serta menjaga kebersihan dan keamanan pantai. Mereka berperan penting dalam menjaga daya tarik dan kenyamanan pantai bagi pengunjung. Saat ini, Pantai Olo terus menjadi tempat yang populer bagi warga Medan dan sekitarnya. Meskipun fasilitasnya mungkin tidak sekomprehensif pantai wisata besar, pesona alam dan suasana tenang di Pantai Olo tetap menarik bagi mereka yang mencari tempat bersantai dan memancing untuk saat ini. Pantai Olo mencerminkan hubungan erat antara masyarakat dan lingkungan alam mereka, serta bagaimana tempat-tempat sederhana dapat menjadi bagian penting dari kehidupan sosial dan budaya sebuah komunitas.

2.2 Sampah

Sampah merupakan masalah terbesar dari semua negara termasuk Indonesia. Jumlah dan jenis sampah di Indonesia terus meningkat setiap tahunnya seiring dengan pertumbuhan jumlah penduduk. Berdasarkan data Sistem Informasi Pengelolaan Sampah Nasional (SIPSN) yang dikutip dari situs resmi SIPSN, jumlah sampah yang dihasilkan di Indonesia pada tahun 2021 sebesar 24,67 juta ton per tahun, yaitu 13. Itu berarti pengurangan sebesar 38% atau 3.3. Namun sejauh ini pengolahan sampah di Indonesia baru terkelola sebesar 50,43% atau 124.444.400.000 ton/tahun. Dengan jumlah tersebut, Indonesia bisa memproduksi sekitar 67.590 ton atau 0,25 kg per orang per hari.

Dari data tersebut, kita dapat menyimpulkan bahwa Indonesia saat ini berada dalam darurat sampah. Ini adalah masalah besar dan tantangan bersama dalam menangani sampah. Undang-Undang Republik Indonesia No. 18 Tahun 2008 tentang Pengelolaan Sampah. Pasal 1(1) menyatakan bahwa “sampah adalah sisa-sisa kegiatan manusia sehari-hari atau gejala alam yang berbentuk padat”. Ada dua jenis sampah yaitu sampah anorganik dan sampah organik. Sampah anorganik merupakan sampah umum yang tidak dapat terurai, seperti: Contoh: logam/besi, pecahan kaca, plastik. Sampah organik biasanya merupakan sampah busuk seperti sisa makanan, daun-daun berguguran, dan buah-buahan [3]. Keadaan sampah di lingkungan sekitar kita selalu tercampur dan tidak dipisahkan [4]. Hal ini disebabkan kurangnya pengetahuan dan kesadaran masyarakat sekitar dalam melaksanakan kegiatan pengolahan sampah sesuai jenisnya. Pengelolaan yang baik, termasuk pengelolaan sampah, khususnya pengklasifikasian sampah ke dalam kategori organik atau anorganik, B3 bertujuan untuk mencegah timbulnya bau dan penyebaran penyakit [5]. Adanya sistem pengenalan jenis sampah diharapkan dapat meringankan permasalahan yang ada dan tentunya memudahkan pengelolaan sampah untuk diolah kembali atau digunakan kembali. (Abdurahman, Rasidi; Pasaribu, Yolanda Al hidayah; Ziqri, Afzal; Adiharma, Faisal Adhinanta,; 2022)

2.3 Machine Learning

Machine Learning merupakan cabang ilmu dalam bidang Kecerdasan Buatan. Tujuan dari *Machine Learning* adalah untuk melatih mesin dengan sample atau *dataset* yang berkaitan terhadap tugas yang harus dilakukan. Mesin akan mempelajari pola-pola dari *dataset* tersebut dan menghasilkan aturan sendiri. Dengan demikian, mesin dapat mengenali data yang dimasukkan ke dalamnya dengan baik. Secara rinci, *Machine Learning* melibatkan beberapa konsep dan komponen utama, termasuk:

1. *Data Training*: Proses di mana model *Machine Learning* dilatih menggunakan data yang telah diberi label. *Data training* ini berfungsi sebagai "pelajaran" bagi model untuk mengenali pola dan menggeneralisasikan pengetahuan dari data tersebut.
2. *Algoritma*: Algoritma *Machine Learning* adalah metode matematis dan

statistik yang digunakan untuk menggali informasi dari data. Ini termasuk metode seperti regresi, pohon keputusan, naive bayes, *support vector machines*, dan *neural networks*.

3. Fitur dan Vektor: Data *input* pada *Machine Learning* direpresentasikan sebagai fitur atau atribut numerik yang relevan. Fitur ini membantu model dalam mempelajari pola dari data. Data *input* ini sering diwakili dalam bentuk vektor.
4. Proses Pelatihan (*Training Process*): Proses di mana model *Machine Learning* disesuaikan dengan data *training*. Ini melibatkan memperbarui parameter model untuk mengoptimalkan prediksi model sesuai dengan data *training*.
5. Evaluasi dan Validasi: Setelah model dilatih, model tersebut dievaluasi dan divalidasi menggunakan data yang terpisah, yang disebut sebagai data validasi atau data pengujian. Evaluasi ini membantu untuk mengukur sejauh mana model dapat melakukan prediksi yang akurat dan mampu menggeneralisasi pengetahuan dari data yang belum dilihat sebelumnya.
6. Prediksi dan Inferensi: Setelah model dilatih dan divalidasi, model tersebut dapat digunakan untuk melakukan prediksi atau inferensi pada data baru yang belum pernah dilihat sebelumnya.

Machine Learning memiliki berbagai jenis dan pendekatan, termasuk *Supervised Learning* (pembelajaran terbimbing), *UnSupervised Learning* (pembelajaran tanpa pengawasan), dan *Reinforcement Learning* (pembelajaran dengan penguatan). Setiap jenis *Machine Learning* memiliki tujuan dan teknik yang berbeda-beda.

Penerapan *Machine Learning* sangat luas dan dapat ditemukan di berbagai bidang, termasuk pengenalan suara dan gambar, analisis data, rekomendasi produk, deteksi penipuan, pengenalan pola, pemrosesan bahasa alami, dan banyak lagi. Dengan kemajuan teknologi dan ketersediaan data yang melimpah, *Machine Learning* terus berkembang dan menjadi salah satu bidang yang paling penting dalam ilmu komputer.

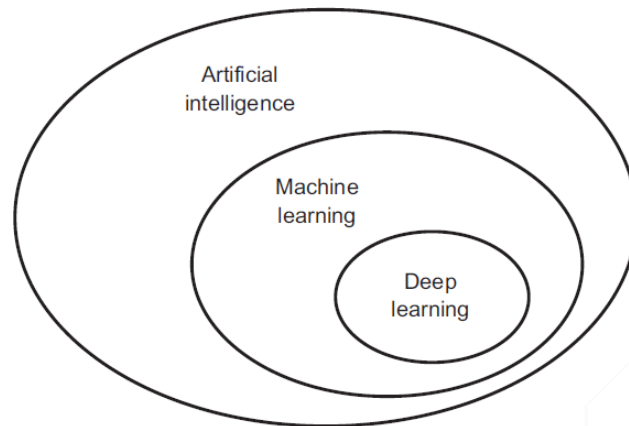
2.4 Deep Learning

Deep Learning adalah cabang dari *Machine Learning* yang merupakan bagian dari Kecerdasan Buatan. *Deep Learning* adalah sebuah metode pembelajaran mesin yang menggunakan jaringan neural yang sangat dalam, terdiri dari banyak lapisan (*layer*), untuk mempelajari representasi yang semakin kompleks dari data yang

Deep learning merupakan metode pembelajaran yang menggunakan jaringan saraf tiruan multilayer. Jaringan saraf tiruan ini mirip dengan otak manusia, dengan neuron-neuron yang terhubung satu sama lain membentuk jaringan saraf yang sangat kompleks. Pembelajaran mendalam atau pembelajaran terstruktur mendalam atau pembelajaran hierarki atau saraf dalam adalah metode pembelajaran yang menggunakan beberapa transformasi nonlinier, dan pembelajaran mendalam dapat dianggap sebagai kombinasi pembelajaran mesin dan AI (jaringan saraf buatan). (Nugroho, Fenriana, & Arijanto, 2020)

Beberapa jenis arsitektur *Deep Learning* yang umum digunakan antara lain *Convolutional Neural Networks* (CNN) yang efektif dalam pengolahan gambar dan pengenalan pola spasial, *Recurrent Neural Networks* (RNN) yang cocok untuk data berurutan seperti teks dan suara, dan *Generative Adversarial Networks* (GAN) yang digunakan dalam pembuatan dan rekonstruksi data baru.

Deep Learning menjadi sangat populer dan sukses karena kemampuannya untuk mempelajari representasi yang kompleks dan abstrak dari data secara end-to-end, tanpa perlu ekstraksi fitur manual. Keberhasilannya dalam berbagai bidang seperti pengenalan wajah, terjemahan mesin, kendaraan otonom, dan banyak lagi, telah membawa kemajuan yang signifikan dalam bidang kecerdasan buatan. Pada Gambar 2.1 akan menunjukkan hubungan antara *Deep Learning*, *Machine Learning* dan kecerdasan buatan.



Gambar 2. 1 Hubungan antara Deep Learning, Machine Learning dan Kecerdasan Buatan (Chollet, 2018)

Neural Network atau Jaringan saraf juga dikenal sebagai jaringan saraf buatan atau hanya saraf buatan, adalah model komputasi yang terinspirasi oleh struktur dan fungsi otak manusia. Ini adalah algoritma pembelajaran mesin yang kuat yang mampu mempelajari pola-pola kompleks dan membuat prediksi berdasarkan data masukan.

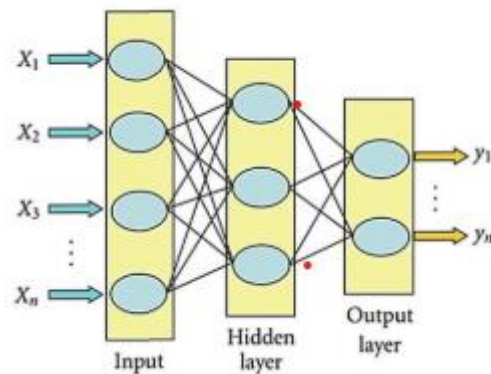
Pada intinya, jaringan saraf terdiri dari simpul yang saling terhubung, atau neuron buatan, yang diorganisir menjadi lapisan. Tiga jenis lapisan utama dalam jaringan saraf adalah lapisan masukan, lapisan tersembunyi, dan lapisan keluaran. Setiap simpul dalam jaringan menerima data masukan, melakukan operasi matematika padanya, dan mengirimkan hasilnya ke lapisan berikutnya. Proses ini disebut propagasi maju.

Keunggulan jaringan saraf terletak pada kemampuannya untuk belajar dari data melalui proses yang disebut pelatihan. Selama pelatihan, jaringan menyesuaikan bobot dan bias yang terkait dengan setiap koneksi antara simpul untuk meminimalkan perbedaan antara keluaran yang diprediksi dan keluaran sebenarnya. Optimasi ini biasanya dilakukan menggunakan algoritma yang disebut propagasi balik, yang menghitung gradien parameter jaringan terhadap fungsi kerugian. Jaringan saraf dapat digunakan untuk berbagai tugas, termasuk klasifikasi, regresi, dan pengenalan pola.

2.5 Artificial Neural Network

Artificial Neural Networks (ANN) adalah model komputasi yang terstruktur dan merupakan representative dari fungsi jaringan saraf manusia, ANN

merupakan sistem klasifikasi dengan karakteristik yang mirip dengan jaringan saraf pada otak manusia. Arsitektur jaringan saraf ini memiliki banyak lapisan: lapisan output, lapisan input dan lapisan *hidden layer*. (Aviya, Irfansyah, & Puspita, 2020)



Gambar 2. 2 Multilayered perceptron (Fatehnia & Amirinia, 2018)

Pada Gambar 2.2 Model MLP (*Multilayer Perceptron*) merupakan jaringan saraf tiruan yang terdiri dari neuron dan lapisan yang dihubungkan oleh bobot. Model ini memiliki kemampuan yang baik dalam mendekati fungsi dalam ruang berdimensi tinggi. Kinerja model MLP ditentukan oleh modifikasi bobot (w) antara lapisan dan neuron. Model ini mampu mempelajari fungsi non linier untuk klasifikasi atau regresi. *Hidden layer* atau lapisan tersembunyi adalah lapisan yang terletak di antara lapisan input dan output. *Hidden layer* melakukan transformasi non linier dari input yang dimasukkan ke dalam jaringan. Jumlah *hidden layer* bervariasi tergantung pada fungsi jaringan ANN dan bobot yang terkait. Selain parameter *hidden layer*, terdapat beberapa parameter lain yang mempengaruhi kinerja ANN, yaitu fungsi aktivasi dan *learning rate*. (Dananjaya, Raden Harya; Sutrisno; Fitriady, Syahid, 2020)

Fungsi aktivasi adalah parameter yang menentukan fungsi matematika yang digunakan model *machine learning* untuk mengelola data input sehingga bisa didapatkan nilai output yang diinginkan. Beberapa fungsi aktivasi yang terkenal adalah fungsi *Identity*, Logistic/Sigmoid, Tanh, dan ReLU. *Learning rate* adalah parameter yang digunakan untuk mengontrol kecepatan algoritma memperbarui bobot dari jaringan dengan memperhatikan penurunan nilai *error*

2.6 Convolutional Neural Network

CNN adalah evolusi dari ANN (*Artificial Neural Network*) yang berfokus pada pemrosesan suara, gambar, dan video. sistem yang menggunakan prinsip sel otak manusia yang sama dengan ANN. Algoritma CNN memulai proses pre-processing dengan mengubah ukuran dan noise pada gambar. Dilanjutkan dengan deteksi wilayah minat (ROI) bertanggung jawab untuk menghilangkan background dan mengambil fitur gambar. Pengenalan objek juga bertanggung jawab untuk mengidentifikasi kesamaan fitur objek yang ditemukan pada gambar. (Stephen, Raymond, & Santoso, 2019)

CNN terdiri dari berbagai jenis lapisan, termasuk lapisan *convolution*, lapisan *pooling*, dan lapisan penuh terhubung, yang membedakan CNN dari ANN yang tidak memiliki lapisan ini.. Hierarki fitur dibangun secara otomatis melalui proses pelatihan, dimulai dari fitur sederhana hingga fitur kompleks, memungkinkan CNN untuk memahami dan mengenali pola secara bertahap. Lapisan fully connected di bagian akhir CNN berperan dalam menghubungkan setiap neuron untuk menghasilkan output yang mewakili probabilitas kelas. Pelatihan CNN melibatkan optimisasi parameter, dan representasi fitur diekstrak secara otomatis selama proses ini. CNN juga sering menggunakan *transfer learning*, memanfaatkan model yang telah dilatih pada dataset besar untuk tugas khusus dengan dataset yang lebih kecil.

2.7 Tensor Flow

TensorFlow adalah *open source* library untuk *machine learning* yang di release oleh Google yang mendukung beberapa bahasa pemrograman. Dalam proses transfer learning, tensorflow berperan untuk memproses Inception-v3 untuk di *training* ulang menggunakan data yang baru dan kemudian menghasilkan classifier dengan komputasi yang cepat dan akurasi yang baik. (Aningtiyas, Sumin, & Wirawan, 2020)

TensorFlow menggabungkan aljabar komputasi teknik pengoptimalan kompilasi, mempermudah penghitungan banyak ekspresi matematis dimana

masalahnya adalah waktu yang dibutuhkan untuk melakukan perhitungan. Fitur utamanya meliputi:

1. Mendefinisikan, mengoptimalkan, dan menghitung secara efisien ekspresi matematis yang melibatkan array multidimensional(tensors)
2. Pemrograman pendukung jaringan syaraf dalam dan teknik pembelajaran mesin
3. Penggunaan GPU yang transparan, mengotomatisasi manajemen dan optimalisasi memori yang sama dan data yang digunakan. Tensorflow bisa menulis kode yang sama dan menjalankannya baik di CPU atau GPU.

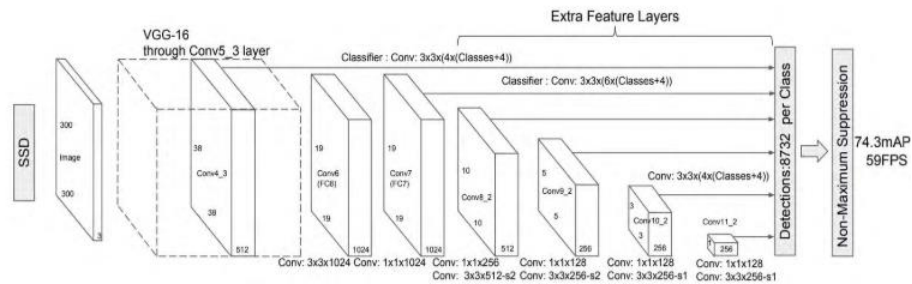
2.8 Single Shot Multibox Detector (SSD)

Single Shot Multibox Detector (SSD) adalah algoritma deteksi objek yang dirancang untuk memberikan performa tinggi dalam mengidentifikasi objek pada citra digital dengan kecepatan tinggi. SSD mencapai akurasi tinggi dan kecepatan real-time dalam deteksi objek melalui beberapa fitur utama. Termasuk SSD-*MobileNet* menggabungkan model *MobileNet* untuk ekstraksi fitur dan model SSD untuk deteksi objek dalam gambar.

MobileNet adalah model berdasarkan arsitektur yang dioptimalkan yang menggunakan konvolusi yang dapat dipisahkan menggunakan *depthwise separable convolutions* dalam membangun *deep neural network*. *depthwise seperable convolution* terdiri dari dua layer yaitu *depthwise convolution* yang berfungsi untuk menerapkan filter pada citra input, dan *pointwise convolution* digunakan untuk menggabungkan hasil output dari *depthwise convolution* sehingga menghasilkan beban komputasi lebih ringan SSD adalah jaringan konvolusional berbasis *feedforward* yang menghasilkan kotak prediktif dan nilai skor prediktif untuk objek citra. SSD menerapkan lapisan fitur konvolusional untuk memprediksi deteksi objek citra. Ukuran lapisan fitur berkurang seiring waktu, memungkinkan Anda melakukan deteksi pada berbagai ukuran. (Syahputra, 2023)

Ukuran fitur adalah ukuran dari setiap fitur yang dihasilkan oleh lapisan konvolusional. Ukuran fitur biasanya dinyatakan dalam format lebar x tinggi. Jumlah kanal adalah jumlah saluran yang digunakan oleh lapisan

konvolusional. Jumlah kanal biasanya dinyatakan dalam format bilangan bulat.



Gambar 2.3 CNN Arsitektur VGG-16 (Ali Rohan., et al., 2020)

Pada Gambar 2.3 Proses ekstraksi fitur pada SSD menggunakan jaringan konvolusional VGG16 sebagai dasar . Jaringan VGG16 terdiri dari 16 lapisan konvolusional. SSD menggunakan lapisan konvolusional terakhir dari jaringan VGG16 untuk menghasilkan fitur untuk mendeteksi objek. Konvolusi dilambangkan dengan tanda bintang (*). Jadi, $Fg=h$ berarti bahwa fungsi F yang dikonvolusikan dengan fungsi g menghasilkan fungsi h . Konvolusi antara dua fungsi $F(x)$ dan $g(x)$ didefinisikan seperti yang ditunjukkan berikut:

$$h(x) = F(x) * g(x) = \int F(a)g(x-a) \quad(1)$$

Untuk fungsi diskrit, konvolusi di definisikan seperti berikut:

$$h(x) = F(x) * g(x) = \sum_{-\infty}^{\infty} F(a)g(x-a) \quad(2)$$

Kernel konvolusi (filter) $g(x)$ adalah sebuah jendela yang digeser sepanjang sinyal masukan $F(x)$. Proses ini menghasilkan keluaran konvolusi yang dinyatakan sebagai $h(x)$. Secara umum, metode *Single Shot Detector (SSD)* memiliki rumus sederhana untuk menentukan default boxes dan skala default boxes. Dalam rumus ini, N adalah jumlah default boxes, L_{conf} adalah loss untuk klasifikasi, L_{loc} adalah loss untuk lokalisasi, L adalah prediction box, dan g adalah truth ground box. Rumus ini digunakan untuk menentukan default boxes.

$$L(x, c, l, g) = \frac{1}{N} (L_{conf}(x, c) + L_{loc}(x, l, g)) \quad(3)$$

Proses ekstraksi fitur pada SSD dapat dijelaskan sebagai berikut:

- Lapisan konvolusional terakhir dari jaringan VGG16 menghasilkan fitur

300x300. Fitur ini disebut sebagai *Feature map*.

- *Feature map* diproses oleh empat lapisan konvolusional yang lebih kecil. Setiap lapisan konvolusional menghasilkan *feature map* dengan ukuran yang lebih kecil.
- *Feature map* dari setiap lapisan konvolusional digunakan untuk menghasilkan prediksi *bounding box*

Jumlah prediksi *bounding box* yang dihasilkan oleh setiap lapisan konvolusional = $4 * 3 * 2 * 2 = 48$

Jumlah total prediksi *bounding box* yang dihasilkan oleh semua lapisan konvolusional = $48 * 4 = 96$

Ket :

- a) Jumlah prediksi *bounding box* yang dihasilkan oleh setiap lapisan konvolusional adalah 24.
- b) Jumlah total prediksi *bounding box* yang dihasilkan oleh semua lapisan konvolusional adalah 96.
- c) Jumlah skala adalah jumlah ukuran *bounding box* yang berbeda yang dihasilkan oleh lapisan konvolusional. Dalam arsitektur SSD, jumlah skala adalah 4..
- d) Jumlah aspect ratio adalah jumlah rasio aspek *bounding box* yang berbeda yang dihasilkan oleh lapisan konvolusional. Dalam arsitektur SSD, jumlah aspect ratio adalah 3.
- e) Jumlah kelas adalah jumlah kelas objek yang akan dideteksi oleh jaringan. Dalam arsitektur SSD, jumlah kelas adalah 21.
- f) Jumlah *bounding box* per skala dan aspect ratio adalah jumlah *bounding box* yang dihasilkan oleh lapisan konvolusional untuk setiap skala dan aspect ratio. Dalam arsitektur SSD, jumlah *bounding box* per skala dan aspect ratio adalah 2

Bounding box adalah kotak persegi panjang yang digunakan untuk melokalisasi objek yang terdeteksi dalam gambar, Pada proses prediksi *bounding box*, jaringan SSD menghasilkan prediksi *bounding box* untuk setiap kemungkinan objek dalam gambar. Prediksi *bounding box* ini berisi informasi tentang posisi, ukuran, dan kelas objek . Proses prediksi *bounding box* pada SSD dilakukan dalam dua langkah:

- Pada langkah pertama, jaringan *SSD* menghasilkan prediksi bounding box untuk setiap kemungkinan objek dalam gambar. Prediksi ini didasarkan pada fitur-fitur dari lapisan konvolusional. jaringan *SSD* menggunakan lapisan konvolusional untuk menghasilkan prediksi bounding box untuk setiap kemungkinan objek dalam gambar. Prediksi bounding box ini berisi informasi tentang posisi, ukuran, dan kelas objek
- Pada langkah kedua, jaringan *SSD* menerapkan *Non-Maximum Suppression* untuk menghilangkan prediksi bounding box yang saling tumpang tindih. Langkah ini memastikan bahwa hanya ada satu prediksi bounding box untuk setiap objek dalam gambar. jaringan *SSD* menerapkan *Non-Maximum Suppression* untuk menghilangkan prediksi bounding box yang saling tumpang tindih. Langkah ini memastikan bahwa hanya ada satu prediksi bounding box untuk setiap objek dalam gambar.

Non-Maximum Suppression (NMS) adalah teknik yang digunakan dalam deteksi objek untuk menyaring prediksi kotak pembatas yang tumpang tindih. Setelah model menghasilkan beberapa prediksi kotak pembatas untuk objek yang sama, NMS diterapkan untuk memilih kotak pembatas yang paling akurat dan relevan. Hal ini dicapai dengan membandingkan *Trust Score* dari *Bounding Box* dengan menekan skor yang memiliki skor lebih rendah, sambil mempertahankan skor yang memiliki skor tertinggi. NMS membantu menghilangkan deteksi duplikat dan menyempurnakan rangkaian kotak pembatas terakhir, meningkatkan akurasi deteksi objek. NMS adalah langkah penting dalam *pipeline* deteksi objek, karena NMS memastikan bahwa hanya prediksi kotak pembatas yang paling yakin dan akurat yang dipertahankan, sekaligus menekan prediksi yang berlebihan atau kurang akurat. Proses ini sangat penting untuk mencapai akurasi tinggi dalam tugas deteksi objek.

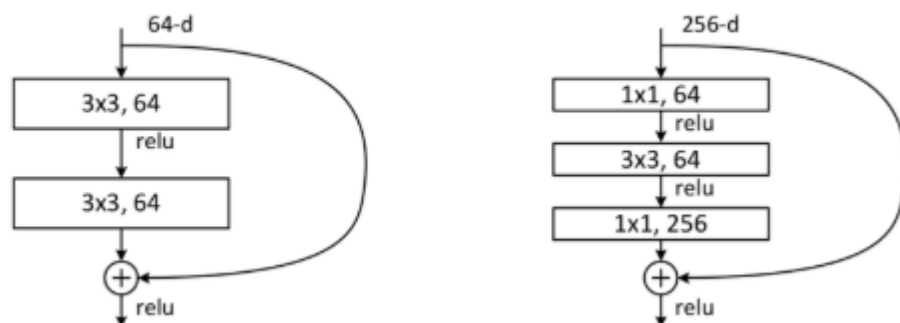
Dalam konteks *Single Shot MultiBox Detector* (*SSD*), NMS digunakan untuk menyaring prediksi kotak pembatas yang berlebihan secara efisien, sehingga berkontribusi terhadap akurasi tinggi dan kecepatan real-time model *SSD* (). Penggunaan NMS di *SSD* memungkinkan pemilihan prediksi kotak pembatas yang paling relevan dan akurat, sehingga menghasilkan peningkatan kinerja deteksi (). (Liu , Anguelov, Erhan, Szegedy, & Reed, 2019).

2.9 Resnet -18

Residual Network (*ResNet*) merupakan salah satu dari deep neural networks yang memiliki performa generalisasi paling baik dalam pengenalan objek. Hal ini didasarkan pada hasil evaluasi menggunakan dataset PASCAL VOC 2007 dan 2012, serta COCO (Me, et al, 2016).

Pada *deep* CNN, semakin besar jumlah layer pada arsitektur maka proses pelatihan menjadi lebih sulit dan akurasi mulai stagnan bahkan menurun. Pembelajaran residual didesain untuk mengatasi masalah penurunan akurasi ini (Ile. et al. 2016). Pembelajaran residual menggunakan metode *shortcut* atau *shortcut connection* untuk menghubungkan input pada beberapa layer berikutnya secara langsung."vanishing gradient" yang sering menghambat jaringan saraf tiruan yang dalam.

ResNet-18, dengan 50 lapisan deep CNN, adalah iterasi dari arsitektur *ResNet* yang memanfaatkan pembelajaran residual. Berbeda dengan pendahulunya yang lebih pendek, *ResNet-18* memperkenalkan perubahan kecil namun signifikan. Misalnya, jumlah shortcut connections yang sebelumnya melewati 2 lapisan pada *ResNet* sekarang meningkat menjadi 3 lapisan pada *ResNet-18*. Selain itu, ada juga penambahan satu lapisan konvolusi tambahan. Perubahan ini bertujuan untuk meningkatkan kemampuan jaringan dalam mempelajari representasi yang semakin kompleks dan mendalam. Teks



Gambar 2. 4 Blok Residual Learning Resnet (Nofal Anam , 2022)

Pada Gambar 2.4 Blok-blok residual merupakan unit dasar arsitektur *ResNet-18*. Setiap blok terdiri dari dua operasi konvolusi 3x3, diikuti dengan

operasi batch normalization dan *ReLU activation function*.

Di antara blok-blok residual terdapat operasi *shortcut connection* yang menghubungkan input blok dengan output-nya secara langsung. Lebih kompleks dan membutuhkan pemahaman yang lebih mendalam, *Shortcut connection* memungkinkan gradien untuk mengalir langsung dari input blok ke output-nya, tanpa harus melewati semua lapisan konvolusi. Hal ini membantu mengatasi masalah "vanishing gradient" yang sering terjadi pada jaringan saraf tiruan yang dalam. Di bagian kiri gambar, terdapat input gambar yang diproses oleh *ResNet-18*. Di bagian kanan gambar, terdapat output jaringan, yang dapat berupa klasifikasi gambar, deteksi objek, atau segmentasi gambar. *Bottleneck residual block* merupakan jenis blok residual yang menggunakan lapisan konvolusi 1×1 untuk mengurangi jumlah kanal (channel) dalam blok. Hal ini dapat meningkatkan efisiensi blok dan membuatnya lebih tahan terhadap *overfitting*.

layer name	output size	18-layer
conv1	112×112	
conv2_x	56×56	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$
conv3_x	28×28	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$
conv4_x	14×14	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$
conv5_x	7×7	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 2$
	1×1	
FLOPs		1.8×10^9

Gambar 2. 5 Arsitektur ResNet-18, (H. Zayd et al., 2022)

Pada Gambar 2.5 Operasi Konvolusi 3×3 , 64: Melakukan konvolusi pada input dengan kernel 3×3 dan menghasilkan 64 kanal. Fungsi Aktivasi ReLU: Menerapkan fungsi ReLU (*Rectified Linear Unit*) untuk mengubah nilai negatif menjadi nol dan meloloskan nilai positif. Operasi Konvolusi 3×3 , 64 (Kedua): Melakukan konvolusi lagi dengan kernel 3×3 dan menghasilkan 64 kanal. Fungsi Aktivasi ReLU (Kedua): Menerapkan fungsi ReLU lagi untuk mengubah nilai negatif menjadi nol. Operasi Konvolusi 1×1 , 256: Melakukan konvolusi dengan kernel 1×1 untuk mereduksi jumlah kanal menjadi 256. Fungsi Aktivasi ReLU (Ketiga): Menerapkan fungsi ReLU untuk mengubah nilai negatif menjadi

nol. Penjumlahan: Menambahkan output dari operasi konvolusi 1×1 , 256 dengan input awal (peta fitur 64 kanal). Fungsi Aktivasi ReLU (Keempat): Menerapkan fungsi ReLU sekali lagi untuk mengubah nilai negatif menjadi nol. Output: Menghasilkan peta fitur dengan 256 kanal.

Arsitektur jaringan konvolusi (CNN) dengan 18 lapisan. Tabel ini terdiri dari beberapa kolom yang memberikan informasi tentang nama setiap lapisan, ukuran output dari lapisan tersebut, dan konfigurasi kernel yang digunakan dalam setiap lapisan untuk model 18 lapisan.

Lapisan pertama, yang diberi label conv1, menghasilkan output berukuran 112×112 . Selanjutnya, pada conv2_x, outputnya berukuran 56×56 . Pada lapisan ini, terdapat dua blok konvolusi berturut-turut dengan masing-masing menggunakan kernel berukuran 3×3 dan 64 filter, yang diulang dua kali (dilambangkan dengan $\times 2$). Ini berarti ada total empat operasi konvolusi di lapisan conv2_x.

Lapisan berikutnya, conv3_x, menghasilkan output berukuran 28×28 dan terdiri dari dua blok konvolusi dengan kernel 3×3 dan 128 filter, juga diulang dua kali. Ini menunjukkan adanya empat operasi konvolusi dalam lapisan conv3_x. Setelah itu, conv4_x menghasilkan output berukuran 14×14 dengan dua blok konvolusi menggunakan kernel 3×3 dan 256 filter, diulang dua kali, menunjukkan total empat operasi konvolusi.

Lapisan terakhir sebelum *fully connected layers*, conv5_x, menghasilkan output berukuran 7×7 . Lapisan ini terdiri dari dua blok konvolusi dengan kernel 3×3 dan 512 filter, diulang dua kali, sehingga terdapat empat operasi konvolusi. Tabel ini juga mencatat adanya operasi konvolusi tambahan dengan kernel 1×1 .

Selain spesifikasi lapisan, tabel juga mencantumkan jumlah operasi *floating-point* (FLOPs) yang diperlukan untuk keseluruhan model, yaitu 1.8×10^9 . FLOPs adalah metrik yang digunakan untuk mengukur kompleksitas komputasi dari model. Secara keseluruhan, tabel ini memberikan gambaran yang jelas tentang struktur dan kompleksitas komputasi dari model CNN 18 lapisan, menunjukkan bagaimana data diproses dan diubah di setiap lapisan konvolusi

2.10 Roboflow



Gambar 2. 6 Roboflow Image Profile (<https://roboflow.com>)

Roboflow adalah platform end-to-end yang dirancang untuk memudahkan pengembangan aplikasi berbasis visi komputer (computer vision). Platform ini menyediakan alat-alat yang membantu pengguna mengelola seluruh siklus hidup proyek visi komputer, mulai dari pengumpulan data hingga penyebaran model. Dengan Roboflow, pengguna dapat dengan mudah mengunggah gambar, melabeli data, melakukan augmentasi data, melatih model, dan mengimplementasikan model yang telah dilatih. Fitur Utama Roboflow:

- a. **Pengumpulan dan Manajemen Data:** Roboflow memungkinkan pengguna untuk mengunggah gambar dari berbagai sumber, termasuk dari komputer lokal, kamera, atau sumber eksternal lainnya. Setelah gambar diunggah, pengguna dapat mengatur dan mengelola dataset dengan mudah melalui antarmuka yang intuitif.
- b. **Pelabelan Data:** Salah satu fitur penting Roboflow adalah alat pelabelan data yang canggih. Pengguna dapat menambahkan label pada gambar untuk mendefinisikan objek atau area yang relevan. Alat pelabelan ini mendukung berbagai format anotasi seperti bounding box, polygon, dan keypoint, yang penting untuk berbagai jenis proyek visi komputer.
- c. **Augmentasi Data:** Roboflow menyediakan berbagai teknik augmentasi data untuk memperluas dan memperkaya dataset. Augmentasi data melibatkan transformasi gambar seperti rotasi, skala, flip, dan perubahan warna untuk menciptakan variasi tambahan dari dataset asli. Ini membantu meningkatkan kinerja model dengan membuatnya lebih robust terhadap variasi dalam data.
- d. **Training Model** Platform ini mendukung integrasi dengan berbagai *framework* pembelajaran mesin terkemuka seperti TensorFlow, *PyTorch*,

dan Keras. Pengguna dapat dengan mudah membuat dan melatih model dengan dataset yang telah mereka persiapkan. Roboflow juga menyediakan template model yang siap digunakan untuk berbagai tugas visi komputer, seperti deteksi objek, segmentasi, dan klasifikasi gambar.

- e. **Evaluasi dan Validasi Model:** Setelah model dilatih, Roboflow memungkinkan pengguna untuk mengevaluasi kinerja model mereka menggunakan metrik evaluasi standar. Ini membantu dalam mengidentifikasi kekuatan dan kelemahan model, serta melakukan tuning parameter untuk meningkatkan akurasi dan performa.
- f. **Penyebaran Model:** Roboflow mempermudah penyebaran model ke lingkungan produksi. Pengguna dapat mengintegrasikan model mereka dengan aplikasi yang sedang berjalan melalui *API* yang disediakan, atau mengunduh model dalam format yang sesuai untuk penyebaran di perangkat keras tertentu.

2.11 Penelitian yang Relevan

Berikut beberapa penelitian yang relevan:

- 1) Pada penelitian berjudul “Implementasi *Deep Learning* dengan *Convolutional Neural Network* untuk Klasifikasi gambar sampah organik dan Anorganik” menyampaikan telah berhasil mengidentifikasi sampah baik anorganik dan organik dengan Tingkat akurasi 95% menggunakan Algoritma ANN sebagai acuan dengan menggunakan data training sebanyak 223 gambar dan data testing 55 gambar (Octavia,Putranto et al.,2022).
- 2) Pada penelitian berjudul “Deteksi Jenis Sampah Menggunakan Metode *Singel Shot Multibox* peneliti berhasil membuat model untuk mendeteksi objek sampah organik dan anorganik. Peneliti menggunakan data training sebanyak 200 gambar dan untuk data testing sebanyak 30 gambar. Hasil perhitungan *confusion matrix* dari model yang sudah berhasil dibuat memiliki akurasi sebesar 93% (Muhammad Rafi et al.,2023)
- 3) Pada penelitian yang berjudul “Pembuatan Aplikasi Deteksi Objek

Menggunakan TensorFlow *Object Detection* API dengan Memanfaatkan SSD *MobileNet V2* Sebagai Model Pra-Terlatih” telah berhasil mendeteksi objek dengan menggunakan TensorFlow Object Detection API dengan memanfaatkan SSD *Mobilenet V2* sebagai model pra terlatih. Program dapat mendeteksi 5 kategori kelas objek, yaitu Camera , Handphone , Laptop dan Mouse. Aplikasi dapat menampilkan tingkat pengukuran akurasi masing-masing objek. Hasil pengujian yang dilakukan pada 50 test set dengan jumlah masing masing kelas objek adalah 10 gambar. Gambar objek Camera mendapatkan rata rata presentase sebesar 99%, Handphone sebesar 89.1%, Headphone sebesar 89.1%, Laptop sebesar 89.1%, dan Mouse sebesar 98.8%. sehingga diperoleh rata rata akurasi keberhasilan pendeteksian aplikasi deteksi objek adalah 93.02%.(Prisky Ratna et al.,2020)

- 4) Pada Penelitian yang berjudul “Deteksi Pengenalan Ikan Menggunakan Algoritma *Convolutional Neural Network*“ penelitian ini dilakukan 6 kali percobaan training untuk ditemukan nilai paling baik, dan mendapatkan nilai test score 2.475, test accuracy 0.4237 dan loss sebesar 2.2002. Data yang digunakan pada penelitian ini berupa tangkapan gambar dari hasil video secara langsung/realtime menggunakan webcam. Penelitian ini menghasilkan tingkat akurasi sebesar 85,18% dengan pengujian 27kali yang dimana 4kali tidak dapat mengidentifikasi foto dan 23 kali berhasil dalam mengidentifikasi foto ikan.(R. Mehindra et al.,2020)

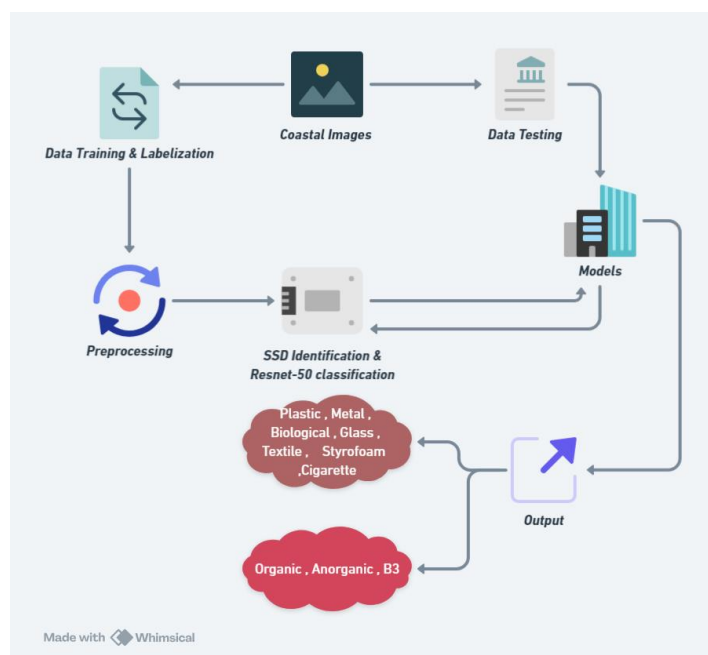
BAB 3

ANALISIS DAN PERANCANGAN

Bab analisis dan perancangan sistem dengan algoritma SSD dan ResNet-18 membahas langkah-langkah yang terlibat dalam menerapkan kombinasi algoritma dalam sistem klasifikasi dan identifikasi sampah. Pada bab ini, pertama-tama dilakukan analisis terhadap kebutuhan dan masalah yang ingin diselesaikan dalam sistem. Hal ini meliputi pemahaman terhadap data yang akan diproses, tujuan pengumpulan data, serta kendala dan batasan yang ada. Setelah itu, dilakukan perancangan sistem dengan mempertimbangkan langkah-langkah yang diperlukan dalam penggunaan, seperti pra-pemrosesan data, penggunaan *embedding layers*, lapisan identifikasi, dan lapisan klasifikasi.

3.1 Arsitektur Umum

Tahapan yang diajukan dalam melakukan kombinasi antara algoritma *SSD* dan ResNet-18 dalam identifikasi dan mengklasifikasikan objek Sampah pesisir terdiri dari beberapa tahapan yang tertera pada Gambar 3.1.



Gambar 3. 1 Arsitektur Sistem

Pada Gambar 3.1 Arsitektur umum sistem melibatkan beberapa langkah penting. Diagram ini menggambarkan proses menyeluruh pelatihan dan pemberian label data untuk klasifikasi gambar menggunakan algoritma *Single Shot MultiBox Detector (SSD)* dan *ResNet-18*. Dimulai dengan akuisisi kumpulan data gambar, yang biasanya terkait dengan tugas klasifikasi spesifik. Kumpulan data ini kemudian dilakukan *preprocessed* terlebih dahulu untuk memastikan konsistensi dan kompatibilitas dengan model. Langkah-langkah praproses meliputi *resizing*, *cropping*, and *normalizing*, serta menambah kumpulan data untuk meningkatkan keragamannya.

Selanjutnya, model *SSD* dan *ResNet-18* dilatih menggunakan kumpulan data yang sudah diproses. Model *SSD* adalah algoritma deteksi objek satu tahap yang secara efisien mendeteksi dan mengklasifikasikan objek dalam gambar. Di sisi lain, model *ResNet-18* adalah jaringan saraf konvolusi residual yang unggul dalam klasifikasi gambar dengan akurasi tinggi. Kedua model dilatih dengan memberi mereka gambar dataset dan label yang sesuai, memungkinkan mereka mempelajari hubungan antara gambar dan kategorinya masing-masing dengan identifikasi akurat. Setelah dilatih, model dievaluasi pada set pengujian terpisah untuk menilai kemampuan generalisasinya yang dibagi menjadi dua output yaitu untuk identifikasi berdasarkan jenis (*Plastic*, *Metal*, *Biological*, *Glass*, *Textile*, *Styrofoam*, *Cigarette*) dan klasifikasi (*Organik*, *Anorganik*, *B3*). Evaluasi ini memastikan bahwa model dapat mengklasifikasikan gambar baru dengan akurat yang belum mereka temui selama pelatihan. Jika model berkinerja baik pada set pengujian, mereka dapat digunakan untuk aplikasi dunia nyata, seperti mengklasifikasikan gambar sampah dari Pantai Olo Medan.

3.2 Labelisasi Dataset

Labelisasi *dataset* dibagi menjadi dua berdasarkan dua algoritma yang merujuk pada proses memberikan label pada data yang menggambarkan jenis dan kategori terhadap suatu entitas atau topik. Dalam topik penelitian label dapat diklasifikasikan ke dalam dua kategori yang nantinya menghasilkan output yang berbeda adalah :

Algoritma SSD menghasilkan Output jenis sampah berdasarkan jenis

1. *Plastic* : Data yang diberi label plastik menunjukkan jenis-jenis sampah plastik yang terdapat pada entitas atau topik yang diamati pada sampah pesisir seperti plastik asoy, plastik minuman kemasan dll
2. *Metal*: Data yang diberi label metal menunjukkan jenis-jenis sampah logam yang terdapat entitas atau topik yang diamati. Contoh jenis sampah botol kaleng, kabel , pipa , seng rumah , baterai, besi dll".
3. *Biological*: Data yang diberi label biologi termasuk jenis sampah yang diperoleh dari makhluk hidup berdasarkan pengamatan terdapat sampah daun kering, buah , ranting pohon, kayu, hewani dll".
4. *Glass* : Data yang diberi label termasuk jenis-jenis sampah kaca seperti pecahan botol , gelas , jendela , cermin
5. *Textile* : Data yang diberikan label textile termasuk jenis sampah hasil olahan mesin yang tidak diinginkan lagi atau dibuang seperti , baju , celana , tas , sandal , kain olahan
6. *Styrofoam* : Data yang diberikan label Styrofoam termasuk bahan yang dibuat dari bahan Styrofoam itu sendiri antara lain tempat makanan , penyimpanan dsb
7. *Cigarette*: Data yang diberikan label rokok termasuk jenis rokok , puntung rokok yang sudah tidak digunakan/dibuang

Algoritma Resnet-18 menghasilkan Output jenis sampah berdasarkan kategori :

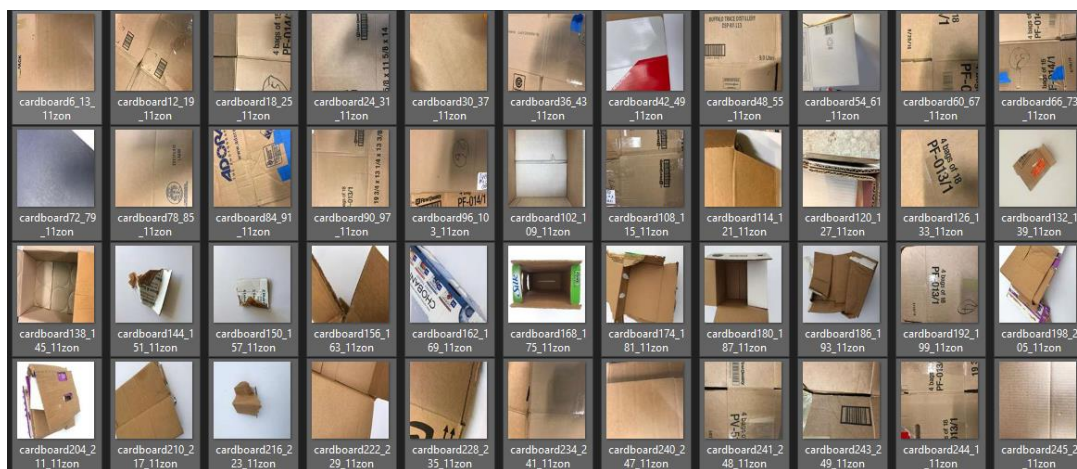
1. Organik : Labelisasi data pada entitas organik meliputi sampah yang dapat di daur ulang dan mudah terurai secara alami meliputi seluruh jenis sampah yang terdapat pada jenis sampah "Biological"
2. Anorganik : Labelisasi data pada entitas anorganik meliputi jenis sampah yang berasal dari bahan-bahan non-hayati, sumber daya alam tidak terbarukan, dan hasil proses teknologi pengelolaan bahan tambang dan industri. Sampah ini umumnya terdiri dari material seperti logam, plastik, kaca, batu, dan textile dan Styrofoam
3. B3 : labelisasi pada entitas B3 meliputi jenis sampah berbahaya yang mengandung zat adiktif/racun yang merusak tubuh umumnya seperti Rokok

Proses labelisasi *dataset* diatas dilakukan secara manual oleh anotator manusia yang mengkategorikan data berdasarkan jenis dan kategori yang terkandung. Juga, algoritma *Machine Learning* dapat digunakan untuk memberikan label secara otomatis dengan mengandalkan model yang telah dilatih sebelumnya menggunakan data yang sudah diberi label oleh manusia. Labelisasi *dataset* dalam hal ini penting untuk dapat mengklasifikasikan dan mengidentifikasi jenis sampah .

3.3 Pengolahan

Persiapan data awal merupakan langkah fundamental dalam identifikasi dan klasifikasi sampah pesisir menggunakan model pembelajaran mesin. Proses ini melibatkan konversi gambar RGB, normalisasi warna, ekstraksi fitur warna, dan pengambilan fitur tekstur.

Langkah pertama adalah mengonversi gambar RGB yang diambil sebagai input menjadi nilai *double* untuk memisahkan kanal warna (Merah, Hijau, Biru). Normalisasi warna kemudian dilakukan pada citra RGB untuk mendapatkan nilai rata-rata normalisasi RGB. Ini membantu menstandarisasi warna gambar dan meningkatkan konsistensi data.



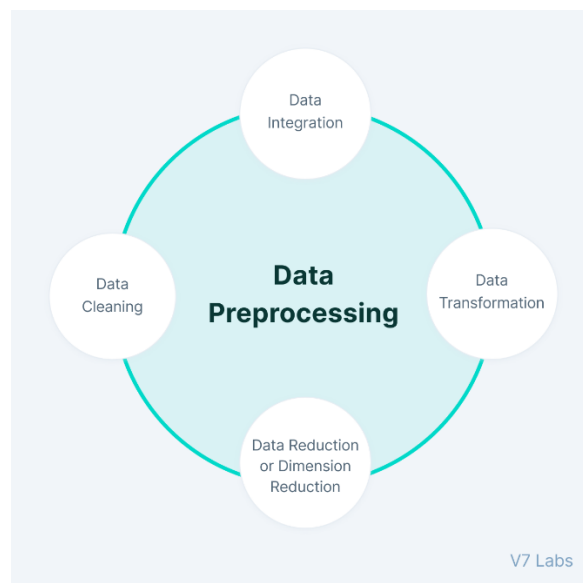
Gambar 3. 2 Sample Gambar

Gambar 3.2 merupakan data yang dikelola atau yang diproses terdiri dari total jumlah 540 data citra setiap class ,pada *SSD* memuat sebanyak 60 data citra pada setiap labelisasi sementara untuk *Resnet-18* masing-masing setiap class memuat 200 data citra Dalam pengambilan fitur tekstur, gambar RGB diubah menjadi

biner. Thresholding diterapkan untuk memisahkan piksel berdasarkan level keabuan mereka, membantu mengidentifikasi batas objek dan area penting dalam gambar. Segmentasi citra atau deteksi tepi kemudian dilakukan untuk meningkatkan tampilan batas objek, membantu model dalam mendeteksi dan mengklasifikasikan sampah pesisir dengan presisi tinggi. Proses pengolahan data awal yang cermat ini menghasilkan representasi gambar yang kaya informasi, siap untuk digunakan oleh model pembelajaran mesin untuk mengidentifikasi dan mengklasifikasikan jenis sampah pesisir dengan presisi tinggi. Hal ini membuka jalan bagi solusi pengelolaan pantai yang lebih efektif dan berkelanjutan. energi, kontras, homogenitas, dan korelasi. Semua tahapan tersebut merupakan bagian dari proses pengambilan nilai tekstur pada gambar

3.4 Pre-processing Dataset

Pre-processing adalah tahap awal dalam pengolahan data yang melibatkan serangkaian teknik untuk membersihkan dan mempersiapkan data mentah sebelum diproses lebih lanjut, seperti penghapusan data duplikat, penghapusan nilai yang hilang, dan transformasi data (Ahmed dan Kamal, 2020). Tahapan - tahapan yang terdapat pada *pre-processing* yang dijelaskan dalam Gambar 3.3.



Gambar 3. 3 Diagram Image Processing

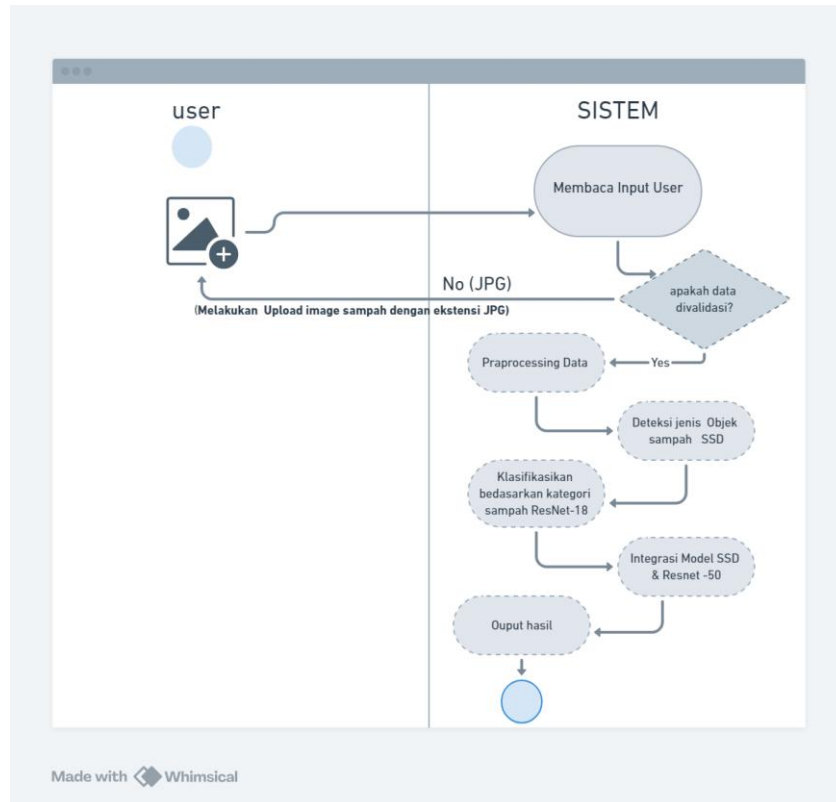
- *Data Integration* mengintegrasikan sumber data untuk menciptakan data set untuk mengidentifikasi dan mengklasifikasikan sampah pesisir secara akurat dengan presisi dan robustitas yang ditingkatkan memungkinkan para peneliti untuk menganalisis pola spasial dan temporal sampah pesisir. Analisis ini dapat mengungkapkan titik panas akumulasi sampah, mengidentifikasi sumber pencemaran, dan melacak efektivitas intervensi pengelolaan sampah.
- *Data Transformation* proses mengubah dan restrukturisasi data mentah agar sesuai untuk digunakan dalam algoritma machine learning. Proses ini meliputi pembersihan, normalisasi, dan rekayasa data untuk meningkatkan kualitasnya, daya prediksi, dan memastikan kompatibilitas dengan model machine learning tertentu. menangani nilai yang hilang Normalisasi data ke rentang yang konsisten Transformasi data ke format yang kompatibel dengan algoritma machine learning
- *Data Reduction* bertujuan untuk meringkas atau memampatkan kumpulan data besar, mengurangi ruang penyimpanan dan meningkatkan kecepatan pemrosesan, tanpa menghilangkan informasi penting. Reduksi dimensi berfokus pada mengurangi jumlah variabel dalam data teknik nya meliputi Sampling: Memilih subset data yang representatif. Agregasi: Menggabungkan titik data yang mirip menjadi satu nilai representatif. Pemilihan fitur: Memilih fitur yang paling relevan untuk tugas yang ada.
- *Data Cleaning* merujuk pada serangkaian proses yang dilakukan untuk membersihkan dan mempersiapkan data sebelum diproses lebih lanjut. Tujuan dari *Data Cleaning* adalah untuk menghilangkan kecacatan, ketidakakuratan, atau noise yang ada dalam data, sehingga data menjadi lebih berkualitas dan siap digunakan untuk analisis atau pemodelan mencakup penghapusan duplikat , *missing values* , outlier ,normalisasi data , validasi data

3.5 Implementasi SSD & Resnet-18

Implementasi model *Single Shot Multibox Detector (SSD)* dengan backbone *ResNet-18* melibatkan beberapa langkah utama. Pertama, pilih *framework* dan pustaka yang sesuai, seperti *Python* dan *TensorFlow/PyTorch*. Selanjutnya, unduh model *ResNet-18 pre-trained* dan implementasikan arsitektur jaringan *SSD*, Dengan *ResNet-18* sebagai *feature extractor*. Definisikan fungsi kerugian dan optimasi yang sesuai, kemudian latih model menggunakan dataset beranotasi dan monitor kinerjanya pada set validasi. Setelah pelatihan selesai, evaluasi model pada set pengujian dan proses penggabungan .

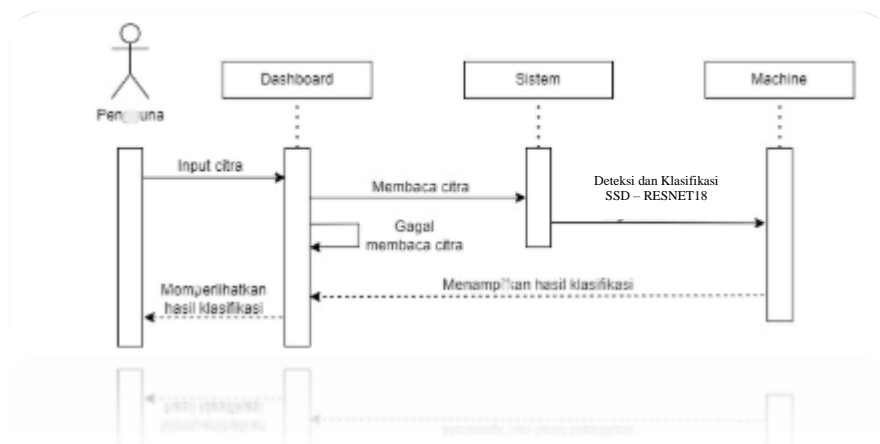
Dalam tahap integrasi model *Single Shot Multibox Detector (SSD)* dan *ResNet-18* melibatkan pengambilan hasil identifikasi dari model *SSD*, yang berupa bounding boxes yang menunjukkan lokasi sampah pesisir pada gambar. Selanjutnya, model *ResNet-18* memberikan hasil klasifikasi berupa jenis sampah yang terdeteksi. Kedua hasil tersebut kemudian digabungkan, memanfaatkan informasi klasifikasi dan lokasi objek, untuk membentuk hasil akhir yang memberikan informasi lengkap tentang sampah pesisir yang terdeteksi. Proses ini dapat melibatkan penggabungan data dan presentasi hasil pada antarmuka pengguna atau sistem informasi, mungkin dalam bentuk peta panas atau gambar yang diberi label. Pengguna atau sistem kemudian dapat mengambil keputusan berdasarkan informasi yang diberikan, seperti memberikan notifikasi atau mengambil tindakan tertentu. Opsionalnya, hasil integrasi dapat dievaluasi kembali untuk memastikan kualitasnya, dan jika diperlukan, dapat dilakukan optimasi atau penyesuaian tambahan pada model. Keseluruhan proses diakhiri dengan langkah penyelesaian yang menandakan bahwa integrasi model *SSD* dan *Resnet* selesai.

3.6 Diagram Alur Sistem



Gambar 3. 4 Diagram Activity

Gambar 3.4 memperlihatkan aksi yang dilalui citra yang pengguna input, citra akan melewati proses dimana jika gambar tidak dalam bentuk ekstensi JPG dikembalikan ke tahap awal. Kemudian jika sistem dapat membaca bentuk file, sistem mendeteksi & mengklasifikasikan sampah terdekat terhadap file image berdasarkan algoritma *Single Shot Multibox Detector (SSD)* dan *ResNet-18*.



Gambar 3. 5 Diagram Sequence

Gambar 3.5 Pengguna memasukkan gambar sampah pesisir ke dalam sistem. Sistem memvalidasi data gambar dan melakukan preprocessing untuk mempersiapkannya untuk proses deteksi dan klasifikasi. Algoritma *SSD* kemudian digunakan untuk mendeteksi objek sampah dalam gambar, menghasilkan kotak pembatas dan bounding box untuk setiap objek. Hasil deteksi ini kemudian diklasifikasikan lebih lanjut menggunakan algoritma *ResNet-18*, yang menentukan kategori sampah dan menampilkan hasil kepada pengguna. mempresentasikan pengguna terhadap *dashboard* ke sistem yang terdapat *machine*. *Machine* mengelola input dan mengklasifikasi output, hasil dari klasifikasi ditampilkan yang dapat dilihat oleh pengguna

3.7 Proses Klasifikasi dan Identifikasi Citra dan perancangan Antarmuka

Proses klasifikasi dan identifikasi citra memanfaatkan algoritma *ResNet-18* dan *Single Shot Multibox Detector* (SSD) untuk mengklasifikasikan dan mengidentifikasi objek dalam citra. Proses ini terdiri dari beberapa langkah penting:

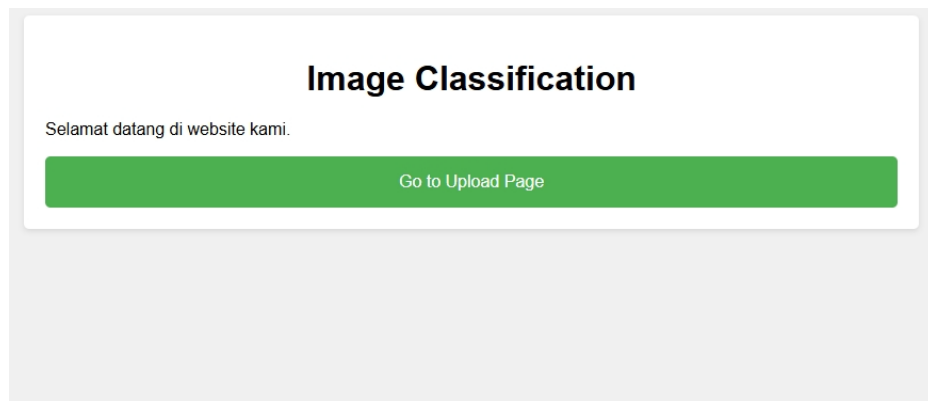
1. Pra-pemrosesan Citra: Sebelum proses klasifikasi dan identifikasi dimulai, citra input mengalami pra-pemrosesan untuk meningkatkan kualitas dan kesesuaiannya dengan algoritma. Hal ini dapat melibatkan teknik pengubahan ukuran, normalisasi, dan pengurangan noise.
2. Ekstraksi Fitur: Algoritma *ResNet-18*, arsitektur jaringan saraf konvolusi (CNN), digunakan untuk mengekstraksi fitur dari citra yang telah dipraproses. CNN mahir dalam mengekstraksi fitur tingkat tinggi dari data visual, yang sangat penting untuk klasifikasi dan identifikasi yang efektif.
3. Deteksi Objek: *Single Shot Multibox Detector* (SSD) memainkan peran penting dalam mendeteksi objek dalam citra. *SSD* memanfaatkan fitur yang diekstrak untuk menghasilkan kotak pembatas di sekitar objek yang terdeteksi, memberikan lokasi objek dalam citra.
4. Klasifikasi Objek: Setelah objek dilokalisasi, algoritma *ResNet-18* digunakan kembali, kali ini untuk mengklasifikasikan objek yang

terdeteksi. Algoritma ini memanfaatkan fitur yang diekstrak untuk menetapkan label kelas ke setiap objek, seperti anorganik , B3 , dan Organik .

Diperlukan Perancangan antarmuka sistem yang ditunjukkan oleh implementasi dalam sistem klasifikasi objek sampah pesisir pada penelitian ini yang terdiri dari beberapa bagian :

3.2.1 Halaman pertama (*landing page*)

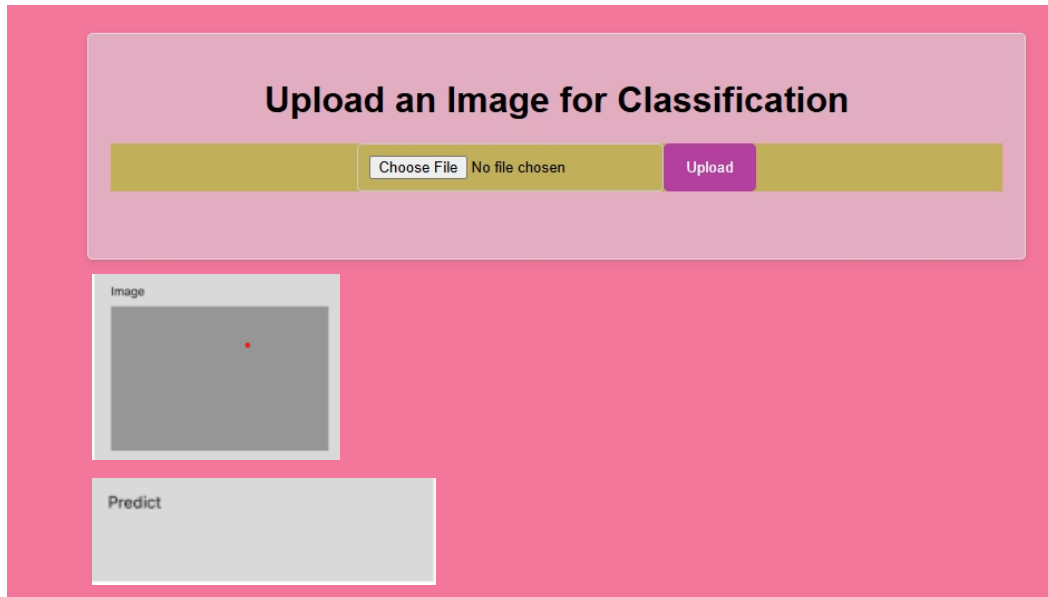
Pada Gambar 3.6 menampilkan *Landing page* menjadi yang pertama dilihat oleh pengguna saat menggunakan website yang dibuat pada penelitian ini.



Gambar 3. 6 *Dashboard Pages*

3.2.2 Halaman hasil uji data (*Upload Pages*)

Pada Gambar 3.7 Halaman ini menjadi hasil klasifikasi data yang sudah di input sebelumnya pada hal pertama data tersebut merupakan citra yang diambil dari data realtime langsung dan kegggle degan resolusi 720p dan hasil prediksi nantinya terdapat lima ouput di algoritma SSD dan 3 output di algoritma Resnet



The screenshot shows a web application interface with a pink background. At the top, there is a light pink box with the title "Upload an Image for Classification". Below the title is a file upload bar with a "Choose File" button, the text "No file chosen", and an "Upload" button. Below the upload bar, there is a section labeled "Image" containing a placeholder image with a red dot in the center. Below the image section, there is a section labeled "Predict" with a large, empty gray box for the prediction result.

Gambar 3. 7 Gambar Halaman Hasil Uji

BAB 4

IMPLEMENTASI DAN PENGUJIAN

Bab implementasi dan pengujian dalam konteks identifikasi dan klasifikasi sampah membahas langkah-langkah yang dilakukan untuk menerapkan model *Singel Shot Multibox Detector* dan *Resnet-18* dalam memprediksi sentimen pada teks atau kalimat. Bab ini menguraikan proses membangun, melatih, dan mengevaluasi model menggunakan kedua Algoritma tersebut

4.1 Implementasi Sistem

4.1.1. Spesifikasi Perangkat Keras

Untuk dapat menerapkan dan mengimplementasikan hasil uji coba model serta melakukan penerapan peneliti melakukan beberapa persiapan dengan spesifikasi laptop sebagai berikut :

1. Processor Inter I5-1100U
2. Memory RAM 8 GB
3. SSD dengan kapasitas 500 GB
4. GPU AMD Radheon Graphic

Untuk memenuhi kebutuhan perangkat keras, penulis memanfaatkan layanan *cloud* gratis dari *Google* yang dikenal sebagai *Google Colab*. Di *Google Colab*, penulis menggunakan GPU, seperti GPU G4, sebagai akselerator perangkat keras untuk membantu dalam proses implementasi dan pengujian. Pendekatan ini diambil karena penulis menggunakan model pra-pelatihan yang berukuran sangat besar.

4.1.2. Spesifikasi Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan dalam penelitian ini adalah sebagai berikut:

1. Sistem Operasi Windows 11 Pro 64 *bit operating system*.
2. *Google Colab*
3. *Python3*
4. *Website RoboFlow (Training Model SSD)*
5. *Library: torch, transformers, numpy, nltk, matplotlib, re, glob , pandas , torchvision , tqdm , sys*

4.2 Implementasi Pengambilan Dataset

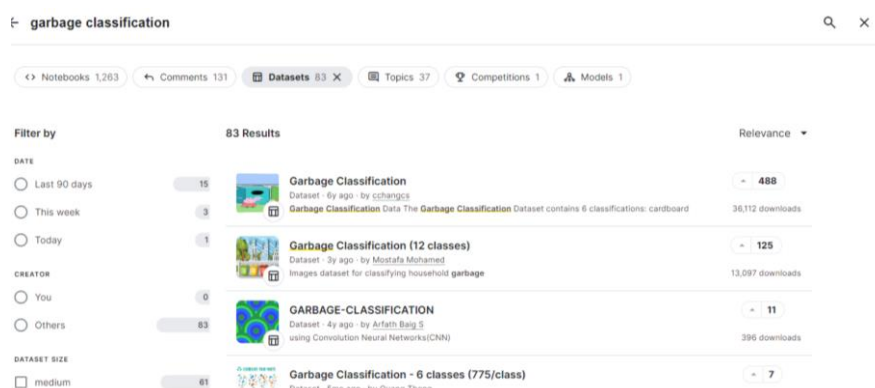
Tahap pertama untuk melakukan implementasi pada pengambilan dataset dilakukan oleh dua cara :

Dataset Real Time yaitu data yang diambil langsung yang di peroleh pada saat mengunjungi studi kasus pada lokasi sampah pesisir dan penulis meng capture beberapa jenis sampah yang terlihat



Gambar 4.1 Jenis Sampah Pesisir

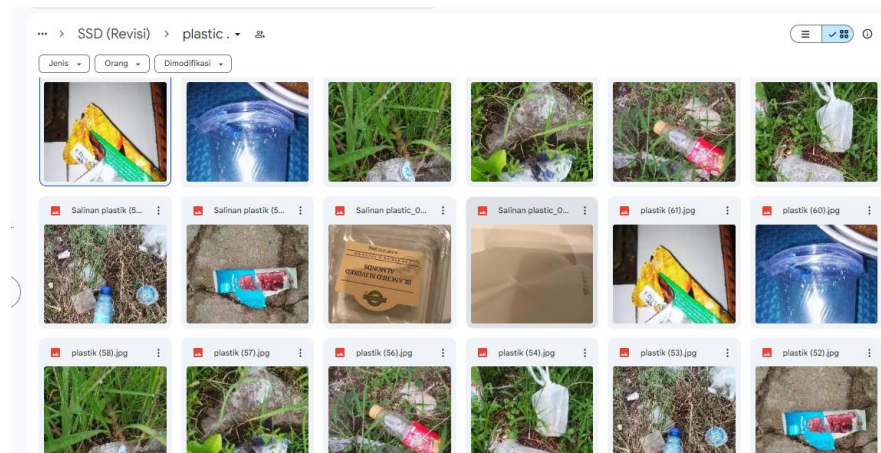
Scrapping Dataset diambil menggunakan website *Kaggle* untuk menambah jumlah data dan mengefesiensi dataset agar dapat berjalan dengan optimal. Penulis membuka halaman *web* pada Website *Kaggle* kemudian penulis melakukan *scrapping* data dengan mendownload salah satu *source* untuk kebutuhan identifikasi dan klasifikasi sampah.



Gambar 4.2 Proses Scrapping dataset kaggle

Dataset diambil berdasarkan setiap jenis class yang sudah disiapkan untuk

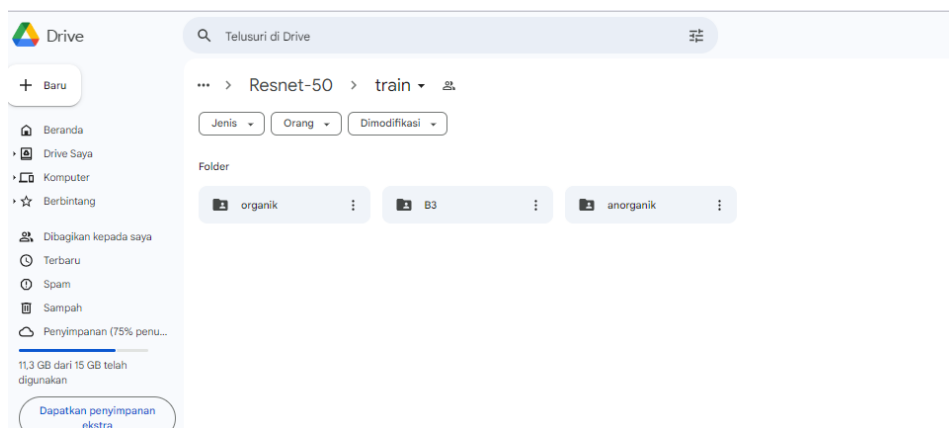
memperoleh output maksimal dan hanya mengambil jenis sampah yang berhubungan dengan peneliti seperti sampah Plastik , Logam , Biologi , Styrofoam, Textile , Rokok dan besarkan klasifikasi Organik , B3 dan Anorganik dan diperoleh Total 500 Data. Seperti pada Gambar 4.3 dataset dikelompokkan menjadi satu class yang di label dan memuat masing “ jumlah gambar sesuai yang ditentukan.



Gambar 4.3 Dataset Gdrive kategori Plastic

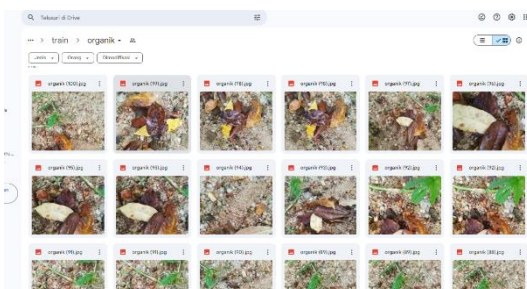
4.3 Labelisasi Dataset

Setelah melakukan pengumpulan data, pelabelan data dilakukan ,file gambar dikumpulkan pada gdrive dan nantinya akan di kelompokkan dan dilakukan Pelabelan data merupakan proses memberikan kategori atau label pada setiap data atau citra yang telah dikumpulkan., Pada resnet pelabelan data dilakukan otomatis dimana setiap citra dikelompokkan menjadi tiga bagian

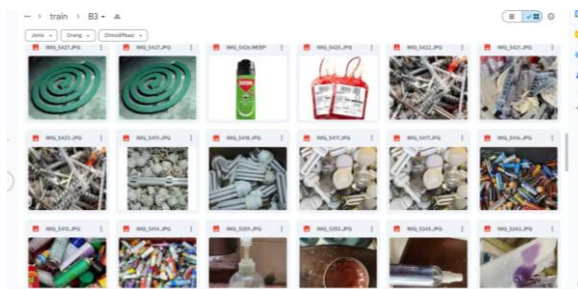


Gambar 4.4 Labelisasi Dataset *Resnet-18*

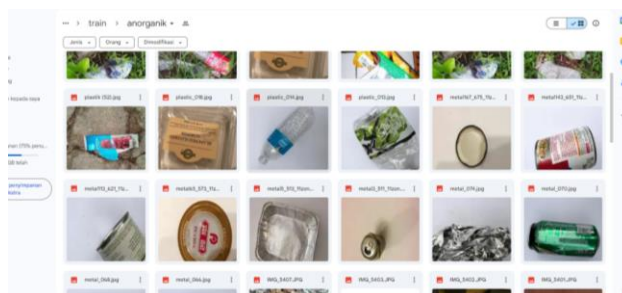
dilihat pada Gambar 4.4 Dataset sample dikelompokkan berdasarkan folder yang sudah ditetapkan dimana data berjumlah 600 dan di split menjadi 200 data pada setiap label yaitu (organik , B3 , Anorganik) pada (Gambar 4.5...Gambar 4.7) masing” label di upload gdrive berbeda. Pelabelan nantinya diberikan secara otomatis menggunakan *syntax* program *python* dan akan dijelaskan di bagian pemodelan nantinya .



Gambar 4.5 Organik

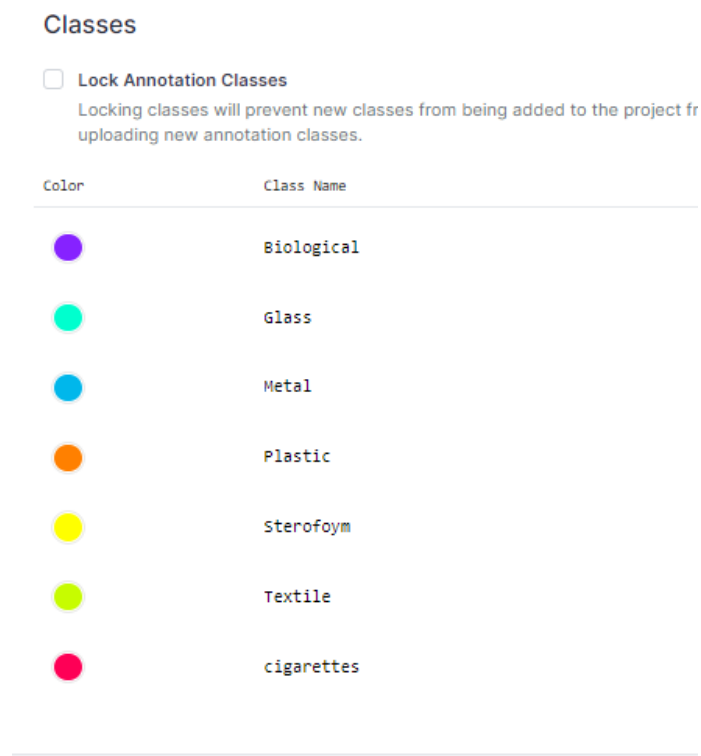


Gambar 4.6 B3



Gambar 4.7 Anorganik

Sedangkan pada labelisasi dataset *SSD* algoritma ini menggunakan website Roboflow , dimana setiap Label terdiri dari 7 class di labelisasi secara manual dengan menggunakan bantuan manusia dimana setiap gambar di bounding berupa kotak persegi pada sekitar objek pada (Gambar 4.8) yang ingin diidentifikasi dan dilakukan *Key Points* Digunakan untuk menandai titik-titik penting pada objek agar dapat meningkatkan akurasi dan presisi saat dilakukan pemodelan dan proses Segmentasi .Masker Digunakan untuk melabeli setiap piksel dalam gambar dengan kelas yang sesuai.dan sesudah di anotasi gambar dimasukan ke label yang sudah dibuat dan disediakan.



Gambar 4.8 Labelisasi Class

(Gambar 4.8) menunjukkan sebuah antarmuka yang berisi dataset yang terdiri dari 420 gambar yang dikelompokkan ke dalam tujuh kategori: *Biological*, *Glass*, *Metal*, *Plastic*, *Styrofoam*, *Textile*, *Cigarette*. Setiap kategori memiliki 60 gambar, yang semuanya telah diberi label. Pada bagian atas antarmuka, terdapat tombol untuk melihat semua 420 gambar secara keseluruhan. Gambar tersebut menggambarkan proses pemberian label data untuk pengenalan objek menggunakan Roboflow. Gambar berlabel ini dapat digunakan untuk melatih model pembelajaran mesin untuk mengidentifikasi dan mengklasifikasikan.

4.4 Preprocessing Dataset

Tahap *preprocessing* pada penelitian ini terdiri dari beberapa proses yaitu *Data Resizing* dan normalisasi yang ada pada Gambar 4.9.

4.4.1. Data Resizing



Gambar 4.9 Data resizing 128 x 128 Resnet

Gambar 4.9 menampilkan proses *Resizing* proses mengubah ukuran gambar ke dimensi tertentu agar memiliki ukuran yang konsisten dimana pada tahap praprocessing Resnet menggunakan Rasio 128 x 128 dan SSD 640 x 640 penggunaan Resizing memastikan bahwa semua gambar memiliki ukuran yang konsisten, yang dapat meningkatkan efisiensi dan akurasi model. Resize dilakukan dengan mempertahankan atau mengubah aspek rasio asli gambar,

4.4.2. Normalisasi

```
# classify using ResNet
def predict_image(img, model, device):
    transform = tt.Compose([
        tt.Resize([128, 128]),
        tt.ToTensor()
```

Gambar 4.10 Transformasi Data

Pada Gambar 4.10 tahap normalisasi *dataset* yang tidak seragam akan diubah menjadi bentuk yang seragam dan dapat diproses dengan mudah. Transformasi ini mengubah gambar yang diubah ukurannya menjadi biner menggunakan tensor *PyTorch*. Tensor adalah struktur data utama dalam *PyTorch* yang digunakan untuk operasi numerik dan komputasi dalam jaringan saraf. Fungsi ini juga mengubah

nilai piksel dari skala [0, 255] (untuk gambar RGB) menjadi skala [0, 1] dengan membagi, dengan (255)

4.5 Split Dataset

Sebelum melakukan identifikasi dan klasifikasi, dataset dibagi menjadi tiga bagian: data latih, data validasi, dan data uji. Data latih digunakan untuk melatih model, sementara data validasi berfungsi untuk mencegah *overfitting* pada model machine learning. Data uji digunakan sebagai uji akhir untuk mengevaluasi seberapa akurat model yang telah dilatih dengan data latih. Berikut adalah ilustrasi tahap pembagian dataset. dapat dilihat pada Gambar 4.9

```
1 | # split dataset into training, testing, and validation
2 | df_train, df_test = train_test_split(df, test_size=0.1)
3 | df_val, df_test = train_test_split(df_test, test_size=0.5)
```

Gambar 4.9 Proses Splitting Dataset

Gambar 4.9 menjelaskan proses data splitting dimana data set di split menjadi tiga bagian yaitu data training , testing dan data valid dimana data dibagi dengan proporsi 90%, 10%, dan 10% masing-masing memastikan model yang dilatih dapat dievaluasi dengan benar menggunakan data yang tidak dilihat selama pelatihan.

4.6 Implementasi Resnet-18

Pertama setiap aspek memiliki datasetnya masing-masing dan dibedakan menjadi data train dan test dimana setiap class berjumlah 200 data train dengan 10% data test pada setiap class kemudian akan dilakukan training pada dataset yang sudah di labelisasi.. Setelah dilakukan training setiap model aspek yang sudah di amount ke drive tadi di panggil ke dalam satu kodingan untuk selanjutnya dilakukan konfigurasi dan lanjut ke tahap pemrosesan Lapisan ini menggunakan teknik Dropout dengan probabilitas 0.1 untuk mencegah *overfitting*. *Fine-tuning* dilakukan dengan menggunakan hyperparameter yang direkomendasikan antara lain :

1. *Epoch*: 10
2. *Batch Size*: 32
3. *Learning rate*: $2e-5$

Pemilihan *Epoch* 10 menyeimbangkan antara performa dan waktu pelatihan, di mana model tidak *underfitting* (kekurangan pelatihan) atau *overfitting* (terlalu detail) terhadap data. *Batch Size* 32 dipilih untuk mencapai keseimbangan antara efisiensi dan stabilitas pelatihan, sehingga prosesnya tidak terlalu lambat atau tidak stabil. *Learning rate* $2e-5$ membantu mencegah *catastrophic forgetting*, di mana model melupakan pengetahuan lama saat mempelajari hal baru, yang penting dalam klasifikasi berbagai jenis sampah.

```

Epoch [0], val_loss: 0.7485, val_acc: 0.7167
Epoch [1], val_loss: 0.7497, val_acc: 0.7333
Epoch [2], val_loss: 0.7499, val_acc: 0.7333
Epoch [3], val_loss: 0.7504, val_acc: 0.7167
Epoch [4], val_loss: 0.7498, val_acc: 0.7167
Epoch [5], val_loss: 0.7490, val_acc: 0.7333
Epoch [6], val_loss: 0.7489, val_acc: 0.7333
Epoch [7], val_loss: 0.7480, val_acc: 0.7333
Epoch [8], val_loss: 0.7487, val_acc: 0.7333
Epoch [9], val_loss: 0.7477, val_acc: 0.7333

history += fit(10, 2e-5, model, train_loader, val_loader)

Epoch [0], val_loss: 0.7477, val_acc: 0.7333
Epoch [1], val_loss: 0.7477, val_acc: 0.7333
Epoch [2], val_loss: 0.7477, val_acc: 0.7333
Epoch [3], val_loss: 0.7477, val_acc: 0.7333
Epoch [4], val_loss: 0.7477, val_acc: 0.7333
Epoch [5], val_loss: 0.7478, val_acc: 0.7333
Epoch [6], val_loss: 0.7479, val_acc: 0.7333
Epoch [7], val_loss: 0.7478, val_acc: 0.7333
Epoch [8], val_loss: 0.7477, val_acc: 0.7333
Epoch [9], val_loss: 0.7477, val_acc: 0.7333

Epoch [0], last_lr: 0.00263, train_loss: 1.2922, val_loss: 1.5213, val_acc: 0.4833
Epoch [1], last_lr: 0.00751, train_loss: 1.2300, val_loss: 1.6103, val_acc: 0.4333
Epoch [2], last_lr: 0.01000, train_loss: 1.3100, val_loss: 1.4592, val_acc: 0.5500
Epoch [3], last_lr: 0.00950, train_loss: 1.3046, val_loss: 1.4398, val_acc: 0.4500
Epoch [4], last_lr: 0.00812, train_loss: 1.2770, val_loss: 1.3364, val_acc: 0.6000
Epoch [5], last_lr: 0.00611, train_loss: 1.2909, val_loss: 1.2891, val_acc: 0.5667
Epoch [6], last_lr: 0.00389, train_loss: 1.2676, val_loss: 1.3913, val_acc: 0.3167
Epoch [7], last_lr: 0.00188, train_loss: 1.2507, val_loss: 1.2514, val_acc: 0.6833
Epoch [8], last_lr: 0.00050, train_loss: 1.2378, val_loss: 1.2438, val_acc: 0.7000
Epoch [9], last_lr: 0.00000, train_loss: 1.2302, val_loss: 1.2443, val_acc: 0.7333
CPU times: user 8min 12s, sys: 29.4 s, total: 8min 42s
Wall time: 9min 43s

[59] model.unfreeze()

[60] %time
epochs = 10
history += fit_one_cycle(epochs, 0.001, model, train_loader, val_loader,
                        grad_clip=grad_clip,
                        weight_decay=weight_decay,
                        opt_func=opt_func)

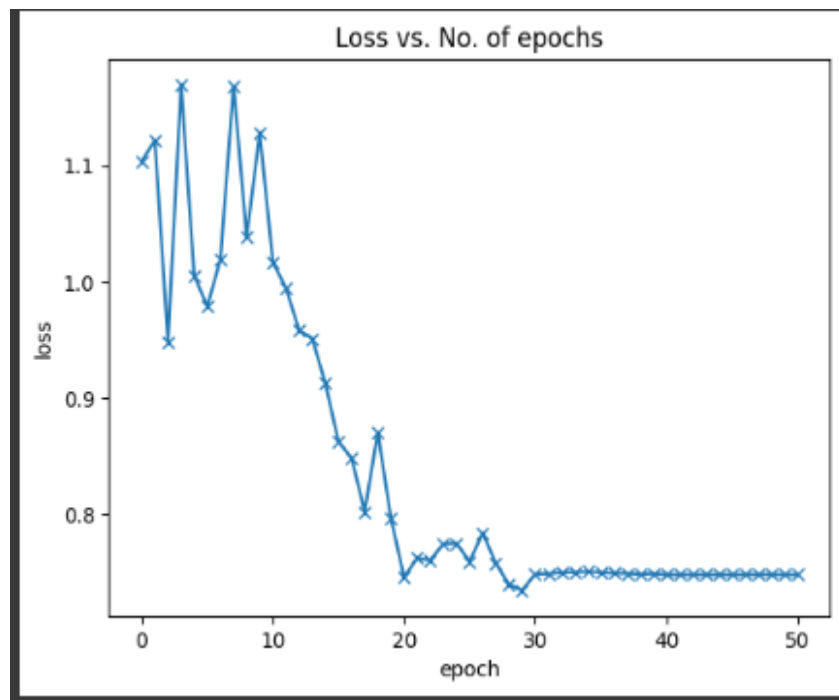
Epoch [0], last_lr: 0.00026, train_loss: 1.2291, val_loss: 1.2522, val_acc: 0.7333
Epoch [1], last_lr: 0.00075, train_loss: 1.2207, val_loss: 1.2244, val_acc: 0.7333
Epoch [2], last_lr: 0.00100, train_loss: 1.2233, val_loss: 1.2834, val_acc: 0.6167
Epoch [3], last_lr: 0.00095, train_loss: 1.2430, val_loss: 1.2276, val_acc: 0.6500
Epoch [4], last_lr: 0.00081, train_loss: 1.2258, val_loss: 1.4964, val_acc: 0.5167
Epoch [5], last_lr: 0.00061, train_loss: 1.1944, val_loss: 1.2030, val_acc: 0.8167
Epoch [6], last_lr: 0.00039, train_loss: 1.2093, val_loss: 1.2126, val_acc: 0.8500
Epoch [7], last_lr: 0.00019, train_loss: 1.1913, val_loss: 1.1918, val_acc: 0.8167
Epoch [8], last_lr: 0.00005, train_loss: 1.1939, val_loss: 1.1886, val_acc: 0.8333
Epoch [9], last_lr: 0.00000, train_loss: 1.1833, val_loss: 1.1917, val_acc: 0.8167
CPU times: user 8min 12s, sys: 30.6 s, total: 8min 42s
Wall time: 9min 43s

```

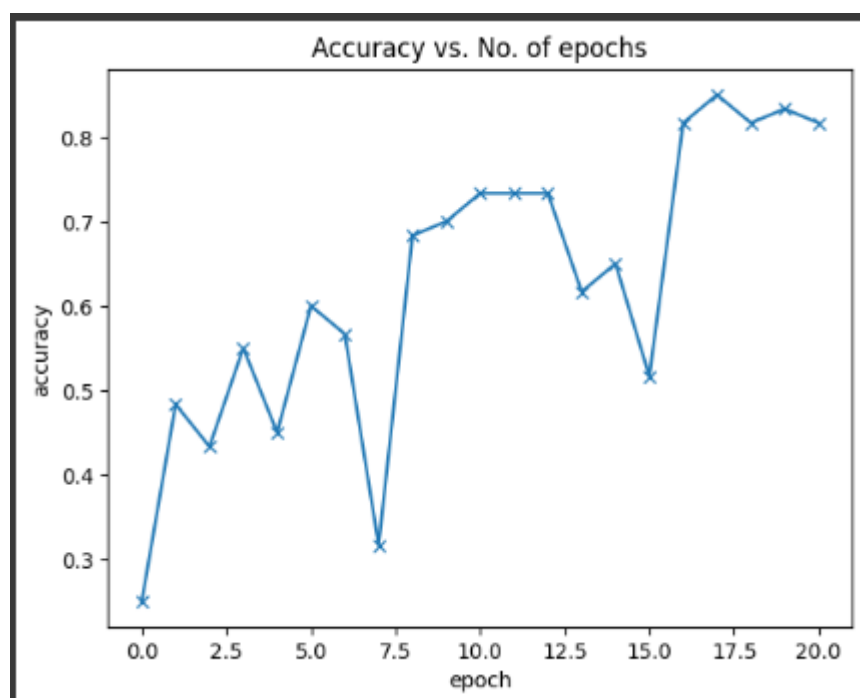
Gambar 4.11 Akurasi Diperoleh pada 5 kali Loop Pelatihan dengan 10 Epoch

Berikut adalah nilai akurasi yang diperoleh saat menggunakan *Epoch* 0 terlihat pada (Gambar 4.11) proses pelatihan melibatkan iterasi atas data pelatihan dalam batch, memasukkan data ke dalam model, menghitung *loss*, dan memperbarui bobot model menggunakan optimizer. Data validasi digunakan untuk memantau performa model selama pelatihan dan mencegah *overfitting*. Cuplikan kode `fit(10, 2e-1, model, train_loader, val_loader)` kemungkinan besar mendefinisikan sebuah loop pelatihan kustom dalam PyTorch untuk melatih model klasifikasi gambar. Kode ini menggunakan learning rate 0.2

dan melatih model selama 50 epoch. Dengan tingkat akurasi 0,81



Gambar 4.12 Grafik Loss setiap Epoch

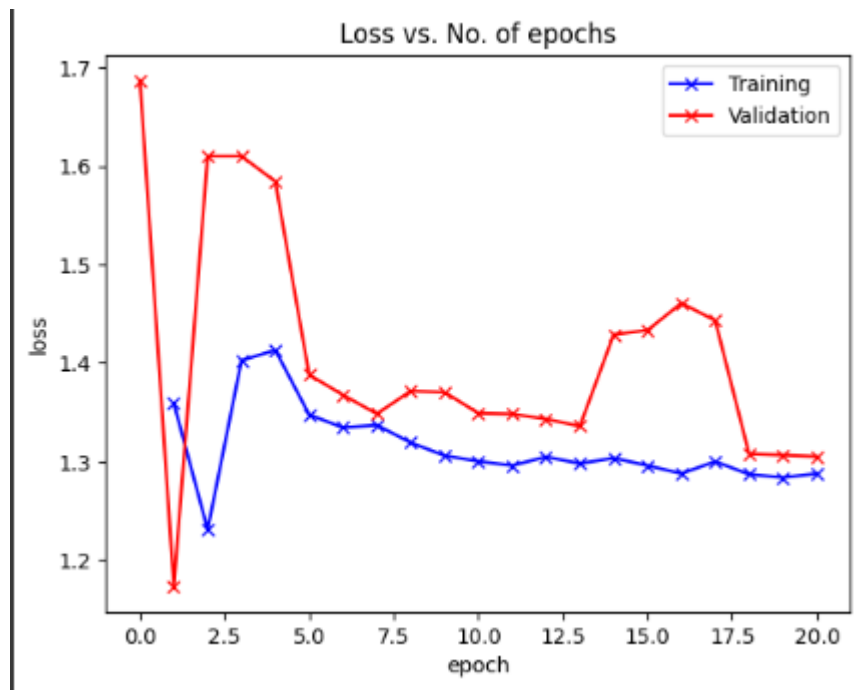


Gambar 4.13 Nilai Accuracy setiap Epoch

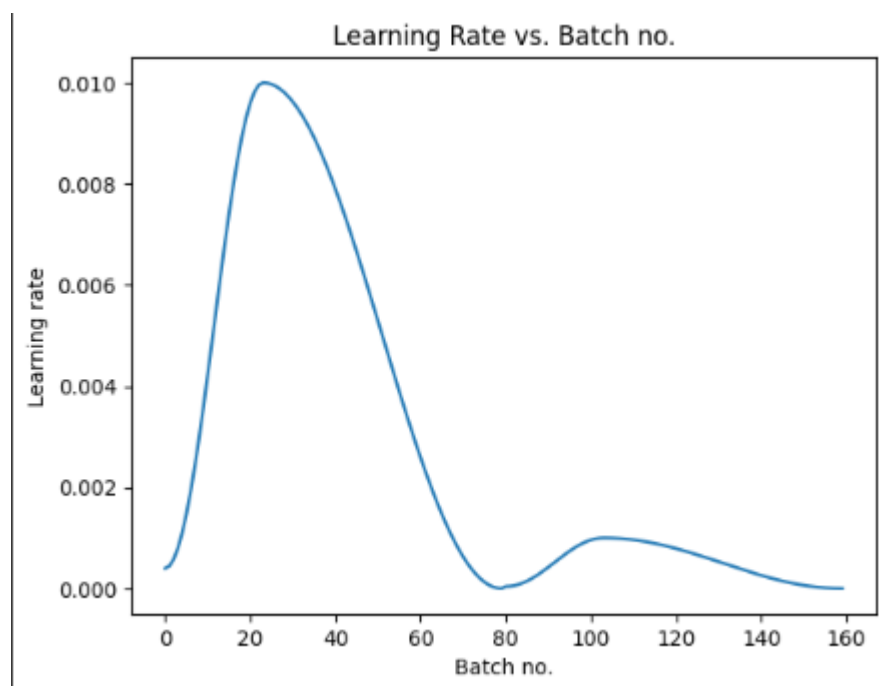
Pada gambar 4.12 Grafik yang ditampilkan menunjukkan hubungan antara jumlah epoch dan nilai kerugian (*loss*) selama proses pelatihan model. Pada sumbu X, terlihat jumlah *epoch*, yang merupakan iterasi pelatihan penuh pada seluruh dataset, sedangkan sumbu Y menunjukkan nilai *loss*, sebuah metrik yang

mengukur seberapa baik model memprediksi data, di mana nilai yang lebih rendah menandakan kinerja model yang lebih baik. Pada fase awal pelatihan, yakni dari *epoch* 0 hingga 10, nilai loss terlihat cukup berfluktuasi. Hal ini menandakan bahwa model masih dalam tahap awal pembelajaran, dengan parameter yang belum optimal. Namun, setelah sekitar *epoch* ke-10, nilai loss mulai menunjukkan penurunan yang lebih konsisten, menandakan bahwa model semakin efektif dalam belajar dan mengoptimalkan parameternya.

Selanjutnya, dari *epoch* 10 hingga 20, penurunan nilai loss masih berlanjut namun dengan laju yang lebih lambat dibandingkan fase sebelumnya, mengindikasikan bahwa model mulai mendekati titik optimalnya. Pada periode *epoch* 20 hingga 30, penurunan loss masih terjadi namun tidak secepat sebelumnya, dan setelah *epoch* ke-30 hingga 50, nilai loss cenderung stabil dan mendatar. Hal ini menunjukkan bahwa model telah mencapai konvergensi, di mana pembelajaran tambahan tidak lagi memberikan penurunan yang signifikan pada *loss*. sebaliknya pada (Gambar 4.13) Pada periode *epoch* 20-30, peningkatan akurasi masih berlanjut meskipun dengan laju yang lebih lambat, mengindikasikan bahwa model mulai mendekati performa optimalnya. Dari *epoch* ke-30 hingga ke-50, akurasi cenderung stabil dan mendatar di sekitar 80%, menandakan bahwa model telah mencapai konvergensi dan pelatihan tambahan tidak lagi memberikan peningkatan signifikan pada akurasi. Tidak ada tanda-tanda overfitting yang terlihat karena nilai loss tidak meningkat kembali setelah stabilisasi, menandakan bahwa model telah menemukan parameter optimalnya.



Gambar 4.14 Proses Training dan Evaluasi Dataset



Gambar 4.15 Perbandingan *learning rate* dan batch size pada setiap *epoch*

Pada gambar 4.14 Pada fase awal pelatihan (*epoch* 0-5), terlihat fluktuasi tinggi pada nilai loss validasi dan pelatihan, yang mengindikasikan bahwa model masih dalam tahap penyesuaian parameter. Setelah *epoch* ke-5, nilai loss pelatihan menurun secara konsisten dan stabil, sementara nilai *loss* validasi juga menurun meskipun dengan beberapa fluktuasi kecil, menandakan bahwa model mulai

belajar lebih efektif. Namun, pada periode epoch 10-15, nilai loss validasi mengalami ketidakstabilan, dengan kenaikan dan penurunan yang signifikan, yang menunjukkan tantangan dalam generalisasi model terhadap data validasi. Pada akhir pelatihan (epoch 15-20), nilai loss pelatihan dan validasi stabil di sekitar nilai rendah, menunjukkan bahwa model telah belajar dengan baik dan tidak ada tanda-tanda *overfitting* yang jelas. Pada Gambar 4.15 Secara keseluruhan, strategi perubahan learning rate yang ditampilkan dalam grafik ini membantu mengoptimalkan proses pelatihan model machine learning, memastikan model dapat belajar dengan cepat pada awalnya dan kemudian menyempurnakan pembelajaran untuk mencapai kinerja terbaik pada akhirnya.

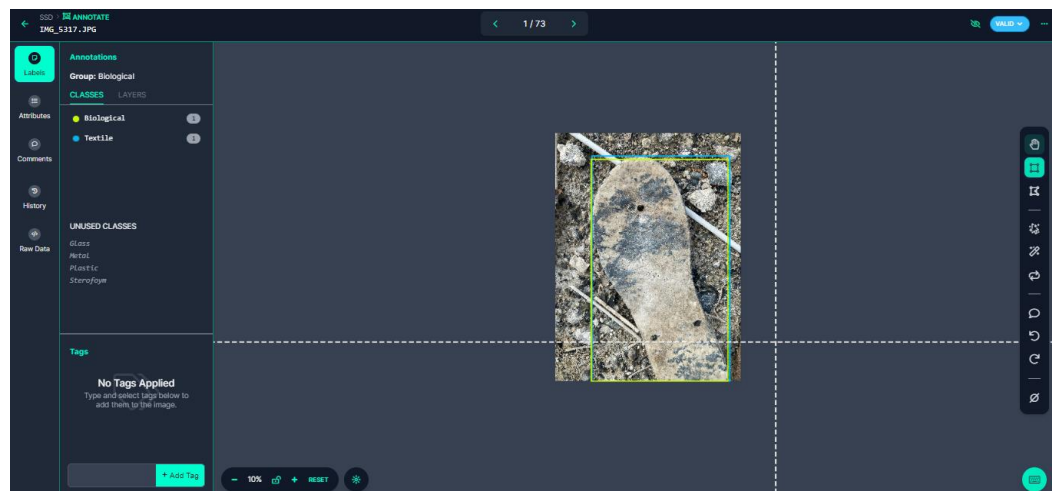


Gambar 4 16 Klasifikasi *Resnet-18*

Gambar 4.16 yang Anda unggah menampilkan hasil dari sebuah model klasifikasi gambar yang telah digunakan untuk mengidentifikasi objek dalam foto. Model ini memprediksi bahwa objek tersebut termasuk dalam kategori "anorganik" dengan tingkat kepercayaan yang sangat tinggi, yaitu 99.9996%. Objek yang ditampilkan dalam gambar adalah bungkus Aluminium Foil merek "DIAMOND". Hasil ini menunjukkan bahwa model klasifikasi berhasil mengidentifikasi aluminium foil sebagai bahan anorganik dengan sangat akurat

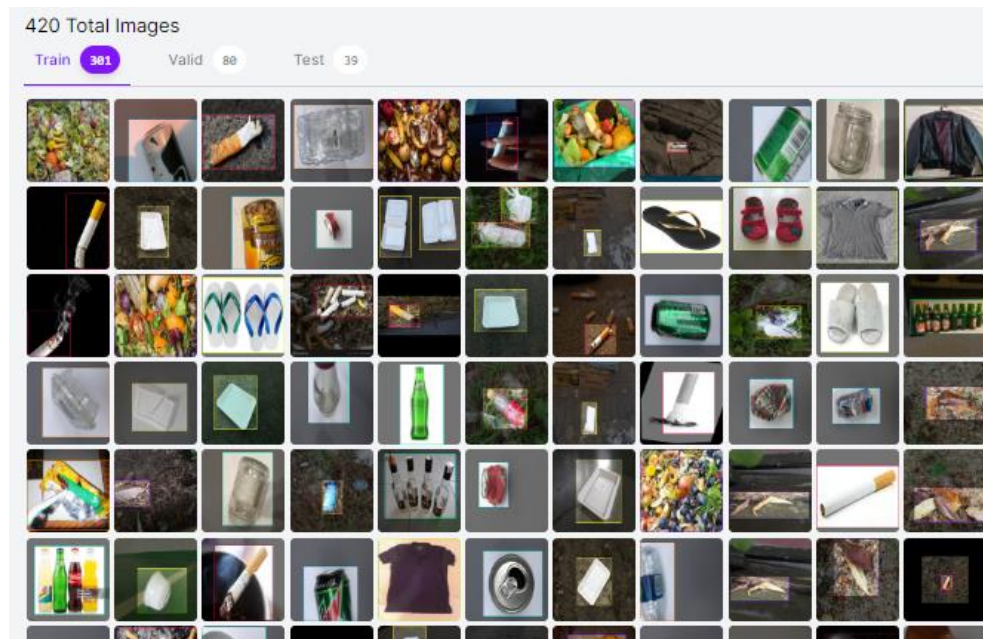
4.7 Implementasi Singel Shot Multibox Detector (SSD)

Pada dasarnya implementasi algoritma SSD penulis menggunakan konfigurasi pada website Tensor Flow guna mempermudah pengerjaan pembuatan model pada sistem identifikasi sampah untuk tahapan pertama dimulai dengan sebagai berikut



Gambar 4.17 Proses Bounding Box SSD

Pada panel sebelah kiri pada Gambar 4.17, terdapat dua label yang telah diterapkan pada gambar, yaitu "Textile", dengan masing-masing ikon berwarna " " dan biru untuk "Textile". Label ini digunakan untuk mengelompokkan objek dalam gambar ke dalam kategori tertentu. Terdapat juga beberapa kelas yang belum digunakan seperti "Glass", "Metal", "Plastic", dan "Styrofoam", "Cigarette". Gambar utama di sebelah kanan menunjukkan objek yang telah diberi kotak pembatas sesuai dengan label yang diterapkan. Di bagian bawah kiri, ada bagian untuk menambahkan tag pada gambar, meskipun saat ini tidak ada tag yang diterapkan. Berbagai ikon alat di sebelah kanan gambar kemungkinan digunakan untuk memperbesar, memperkecil, menggeser gambar, atau mengedit anotasi. Gambar tersebut tampaknya menunjukkan bagian dari sol sepatu yang telah diberi anotasi sebagai "Textile". Proses anotasi gambar ini sangat penting dalam *machine learning* karena memberikan informasi yang diperlukan bagi model untuk belajar mengenali dan membedakan berbagai objek dan kelas dalam gambar. Pemberian label yang tepat serta penggunaan kotak pembatas membantu model untuk memahami dan mendeteksi objek serupa dalam gambar baru.



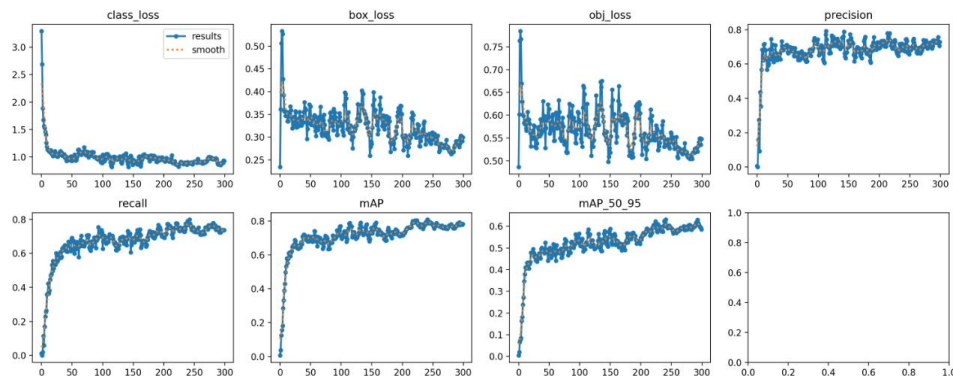
Gambar 4 18 *Training Data* pada dataset Roboflow

Data dibedakan menjadi 7 class pada Gambar 4.18 data di split dan dilakukan bounding pada setiap data gambar yang di peroleh

Data Terdiri dari :

1. *Data Train* : 301 Gambar
2. *Data Valid* : 80 Gambar
3. *Data Test* : 39 Gambar

Setelah data diolah dan dikelompokkan model pun dibuat menggunakan dataset yang sudah di split sebelumnya dengan implementasi Arsitektur *SSD* dan diperolah Grafik pelatihan model



Gambar 4.19 Grafik Pelatihan Model

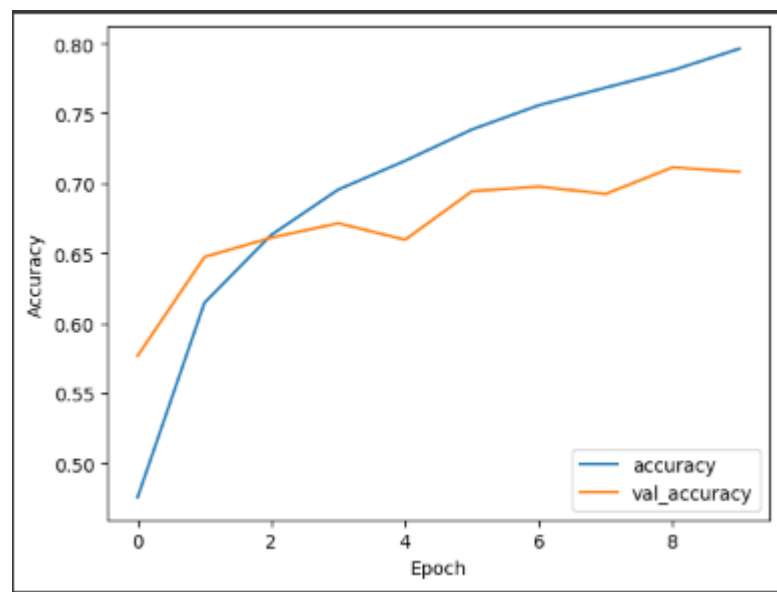
Metrik-metrik ini ditampilkan dalam bentuk plot garis, dengan sumbu x menunjukkan jumlah *epoch* pelatihan (iterasi) dan sumbu y menunjukkan nilai dari masing-masing metrik. Berikut adalah penjelasan dari setiap plot Pada Gambar 4.19 :

1. **box_loss**: Plot ini menunjukkan kehilangan kotak (box loss), yang mengukur akurasi kotak pembatas yang diprediksi di sekitar objek. Kehilangan ini menurun seiring waktu dengan beberapa fluktuasi, menunjukkan perbaikan dalam prediksi kotak pembatas selama pelatihan.
2. **obj_loss**: Plot ini menggambarkan kehilangan objek (*object loss*), yang mengukur kepercayaan model dalam mendeteksi objek. Seperti kehilangan kotak, kehilangan ini menurun seiring waktu dengan beberapa fluktuasi, menunjukkan bahwa kepercayaan model dalam deteksi objek meningkat tetapi bervariasi di berbagai *epoch*.
3. **precision**: Plot ini menunjukkan presisi model, yaitu rasio deteksi benar positif terhadap jumlah total deteksi yang dilakukan. Presisi meningkat tajam pada awalnya dan kemudian stabil, menunjukkan bahwa model semakin presisi dalam prediksinya seiring berjalannya pelatihan.
4. **recall**: Plot ini menggambarkan *recall* model, yaitu rasio deteksi benar positif terhadap jumlah total objek yang sebenarnya. *Recall* meningkat seiring waktu, menunjukkan bahwa model semakin baik dalam mendeteksi semua objek yang relevan.
5. **mAP (mean Average Precision)**: Plot ini menunjukkan mean *Average Precision*, metrik umum untuk mengevaluasi model deteksi objek. Ini

mewakili rata-rata presisi di semua kelas. *mAP* meningkat seiring waktu, menunjukkan peningkatan kinerja keseluruhan model.

6. ***mAP_50_95***: Plot ini menunjukkan *mean Average Precision* pada berbagai ambang *Intersection over Union (IoU)*, mulai dari 0,50 hingga 0,95. Metrik ini memberikan evaluasi yang lebih komprehensif terhadap kinerja model. Nilainya meningkat seiring waktu, menunjukkan bahwa model semakin baik pada berbagai ambang *IoU*.
7. **Plot Kosong**: Plot terakhir kosong, mungkin disediakan untuk metrik lain atau karena kesalahan dalam plot.

```
Epoch 1/10
1563/1563 [=====] - 90s 56ms/step - loss: 1.4555 - accuracy: 0.4755 - val_loss: 1.1942 - val_accuracy: 0.5767
Epoch 2/10
1563/1563 [=====] - 82s 53ms/step - loss: 1.1006 - accuracy: 0.6147 - val_loss: 1.0220 - val_accuracy: 0.6474
Epoch 3/10
1563/1563 [=====] - 82s 52ms/step - loss: 0.9734 - accuracy: 0.6632 - val_loss: 0.9592 - val_accuracy: 0.6612
Epoch 4/10
1563/1563 [=====] - 80s 51ms/step - loss: 0.8825 - accuracy: 0.6956 - val_loss: 0.9515 - val_accuracy: 0.6715
Epoch 5/10
1563/1563 [=====] - 85s 54ms/step - loss: 0.8190 - accuracy: 0.7162 - val_loss: 0.9832 - val_accuracy: 0.6597
Epoch 6/10
1563/1563 [=====] - 81s 52ms/step - loss: 0.7560 - accuracy: 0.7386 - val_loss: 0.8966 - val_accuracy: 0.6944
Epoch 7/10
1563/1563 [=====] - 78s 50ms/step - loss: 0.7055 - accuracy: 0.7559 - val_loss: 0.9025 - val_accuracy: 0.6977
Epoch 8/10
1563/1563 [=====] - 79s 51ms/step - loss: 0.6631 - accuracy: 0.7685 - val_loss: 0.9215 - val_accuracy: 0.6925
Epoch 9/10
1563/1563 [=====] - 79s 51ms/step - loss: 0.6248 - accuracy: 0.7809 - val_loss: 0.8964 - val_accuracy: 0.7115
Epoch 10/10
1563/1563 [=====] - 80s 51ms/step - loss: 0.5812 - accuracy: 0.7964 - val_loss: 0.9156 - val_accuracy: 0.7082
```



Gambar 4.20 Validasi Akurasi SSD

Gambar 4.20 menampilkan Pada epoch ke-10, hasil pelatihan model pembelajaran mesin menunjukkan dua metrik akurasi yang berbeda: akurasi pada data pelatihan dan akurasi pada data validasi. Akurasi pada data pelatihan

mencapai 79.64%, menunjukkan bahwa model mampu mengklasifikasikan 79.64% sampel dalam data pelatihan dengan benar. Sementara itu, akurasi pada data validasi adalah 70.82%, menunjukkan bahwa model mampu mengklasifikasikan 70.82% sampel dalam data validasi dengan benar. jika ingin menghitung "total akurasi" sebagai rata-rata dari kedua metrik tersebut kita bisa melakukan perhitungan sebagai berikut:

$$\text{Total Akurasi} = \frac{79.64 + 70.82}{2} = \frac{150.46}{2} = 75.23\%$$

4.8 Implementasi Tahapan Antar Muka

Implementasi pada tahap antarmuka menggunakan bahasa *Python* untuk pengolahan data di belakang layar, menghubungkan *HTML* sebagai visualisasi antar mukanya. Antarmuka ini terdiri dari halaman pertama dan halaman untuk menampilkan hasil uji data baru. Sebelum dihubungkan model yang sudah dilatih di hubungkan pada dua metode klasifikasi gambar: satu menggunakan model *ResNet* yang dilatih secara lokal, dan satu lagi menggunakan model *SSD* yang disediakan melalui layanan Roboflow.

```

62 model_path = 'my-course-proj-ffn-resnet50_v6.pth'
63 model = IntelImageResnet()
64 model.load_state_dict(torch.load(model_path, map_location=device))
65 model = model.to(device)
66
67 # test_dataset_classes = ['B3', 'anorganik', 'organik']
68 test_dataset_classes = ['B3', 'anorganik', 'organik']
69 # end classify using ResNet
70
71 # classify using SSD
72 rf = roboflow.Roboflow(api_key="Sgz5687lw8IsH1DySWBJ")
73 project = rf.workspace().project("ssd-xyvcj")
74 model2 = project.version("1").model
75 model2.confidence = 20
76 model2.overlap = 10

```

Gambar 4.21 Implementasi dua model Algoritma

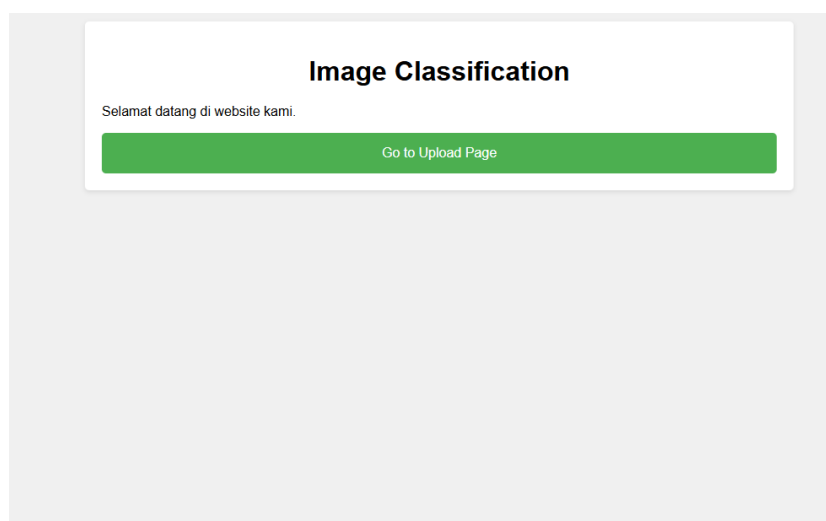
Pada gambar Line 62 – 65 memuat model *ResNet* yang telah dilatih sebelumnya dari file `my-course-proj-ffn-resnet50_v6.pth` menggunakan *GoggleCollab*. Model ini adalah sebuah arsitektur jaringan

saraf yang biasa digunakan untuk tugas klasifikasi gambar. `IntelImageResnet` adalah kelas yang menginisialisasi model, dan `torch.load` digunakan untuk memuat parameter model ke perangkat (device) yang sesuai. Variabel `test_dataset_classes` menyimpan daftar kelas yang akan digunakan untuk pengujian model. Kelas ini adalah label yang akan digunakan untuk mengklasifikasikan gambar-gambar dalam dataset pengujian.

Pada line 72-76 Bagian ini menggunakan layanan dari Roboflow untuk menginisialisasi dan mengkonfigurasi model *SSD*. Roboflow adalah sebuah platform yang menyediakan layanan anotasi dan manajemen dataset untuk pembelajaran mesin. Kode ini mengatur kunci API untuk mengakses proyek yang disebut `ssd-xvycj`, memilih versi model (`version("1")`), dan mengatur parameter *confidence* (kepercayaan) dan *overlap* (tumpang tindih) untuk model tersebut.

4.8.1. Halaman *Dashboard* (Landing Page)

Gambar 4.22 merupakan tampilan utama halaman *dashboard* website



Gambar 4.22 Implementasi Halaman Pertama

4.8.2. Halaman *Upload* (Uji coba data)

Gambar 4.23 menampilkan hasil prediksi sistem untuk data citra yang diuji








Gambar 4.23 Implementasi Halaman Hasil Uji





4.9 Pengujian Sistem






Pengujian sistem klasifikasi dilakukan untuk mengevaluasi kinerja model dalam mengklasifikasikan gambar uji. Melalui pengujian ini, kita dapat menilai seberapa efektif model dalam memprediksi kelas dari gambar yang belum pernah dilihat sebelumnya. Sebelum mengklasifikasikan gambar yang diinput, 16 data digunakan untuk melatih model klasifikasi. Nilai akurasi dihitung dengan membandingkan label kelas yang diprediksi oleh sistem dengan label kelas asli dari gambar uji. Sebelum proses pelatihan, data diberi label dan dilakukan pengolahan citra, hasilnya dapat dilihat pada Table 1.1






Table 1. 1 Pengujian Sistem Implementasi *Website*

No	Gambar Sampah	Hasil Deteksi SSD	Hasil klasifikasi Resnet18	Benar / Salah <input checked="" type="checkbox"/>
1		Textile: <input checked="" type="checkbox"/>	anorganik: <input checked="" type="checkbox"/>	Benar

2		Glass: <input checked="" type="checkbox"/>	anorganik: <input checked="" type="checkbox"/>	Benar
3		Styrofoam <input checked="" type="checkbox"/>	anorganik <input checked="" type="checkbox"/>	Benar
4		Biological: <input checked="" type="checkbox"/>	organik <input checked="" type="checkbox"/>	Benar
5		Biological <input checked="" type="checkbox"/>	Organik <input checked="" type="checkbox"/>	Benar
6		Cigarette <input checked="" type="checkbox"/>	B3 <input checked="" type="checkbox"/>	Benar

7		Biological <input checked="" type="checkbox"/>	Organik <input checked="" type="checkbox"/>	Benar
8		Metal <input checked="" type="checkbox"/>	Anorganik <input checked="" type="checkbox"/>	Benar
9		Biological <input checked="" type="checkbox"/>	Organik <input checked="" type="checkbox"/>	Benar
10		Metal <input checked="" type="checkbox"/>	Anorganik <input checked="" type="checkbox"/>	Benar

11		Cigarette <input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> B3	Benar
12		Textile <input checked="" type="checkbox"/>	Anorganik <input checked="" type="checkbox"/>	Benar
13		Textile <input checked="" type="checkbox"/>	Anorganik <input checked="" type="checkbox"/>	Benar
14		Plastic <input checked="" type="checkbox"/>	Organik <input type="checkbox"/>	Salah
15		Plastic <input checked="" type="checkbox"/>	Anorganik <input checked="" type="checkbox"/>	Benar

16		Plastic <input checked="" type="checkbox"/>	Anorganik <input checked="" type="checkbox"/>	Salah
17		Glass <input checked="" type="checkbox"/>	Anorganik <input checked="" type="checkbox"/>	Benar
18		Glass <input checked="" type="checkbox"/>	Anorganik <input checked="" type="checkbox"/>	Benar
19		Styrofoam <input checked="" type="checkbox"/>	anorganik <input checked="" type="checkbox"/>	Benar
20		Textile <input checked="" type="checkbox"/>	B3 <input checked="" type="checkbox"/>	Salah

Berdasarkan pengujian sistem Table 1.1 menggunakan model kedua algoritma ,diberi data uji coba gambar yang dikelola pada dataset pengujian,. Dataset pengujian terdiri dari 20 sampel dengan dua output identifikasi jenis sampah dan klasifikasi jenis sampah). Dari hasil pengujian, dapat dilihat bahwa

- Pada pengujian 1-13 , 15 ,16 , 17 , 18 , 19 Algoritma dapat mengidentifikasi dengan baik data jenis sampah yang di lihat dengan uji coba Hasil ini menandakan tingkat akurasi memberikan performa terbaik dalam konteks pengujian sample.
- Table 16 *SSD* yang tidak mampu mengidentifikasi jenis sampah metal(logam) dengan baik . Hal ini disebabkan oleh kondisi dataset yang kurang beragam baik dari pengambilan sudut pandang menyebabkan algoritma salah mengidentifikasi sample gambar
- Table 14 terjadi kesalahan karena *Resnet-18* menilai banyak nya rumput dan dedaunan menyebabkan proses output menghasilkan output organik .
- Table 20 kesalahan terjadi karena model tidak dilatih dengan cukup banyak contoh dari berbagai jenis sarung tangan atau benda tekstil lain yang serupa, hal ini bisa membuat model tidak mampu mengidentifikasi gambar dengan benar. Sarung tangan yang ada di gambar mungkin memiliki bentuk atau fitur visual yang menyerupai objek lain yang sudah dikenali model sebagai B3 (Bahan Berbahaya dan Beracun). Hal ini bisa menimbulkan kesalahan dalam identifikasi

Akurasi hasil pengujian sistem klasifikasi citra digunakan untuk menilai kinerja model klasifikasi. Akurasi ini dihitung dengan membandingkan jumlah citra yang diklasifikasikan dengan benar dengan jumlah total citra dalam dataset uji. Semakin tinggi nilai akurasi yang diperoleh, semakin baik performa model klasifikasi.

$$\text{Accuracy} = \frac{\text{TP}}{\text{Tinstance}} = \frac{17}{20} = 0.85 = 85\%$$

Nilai Total prediksi adalah nilai dari semua sample yang bernilai benar

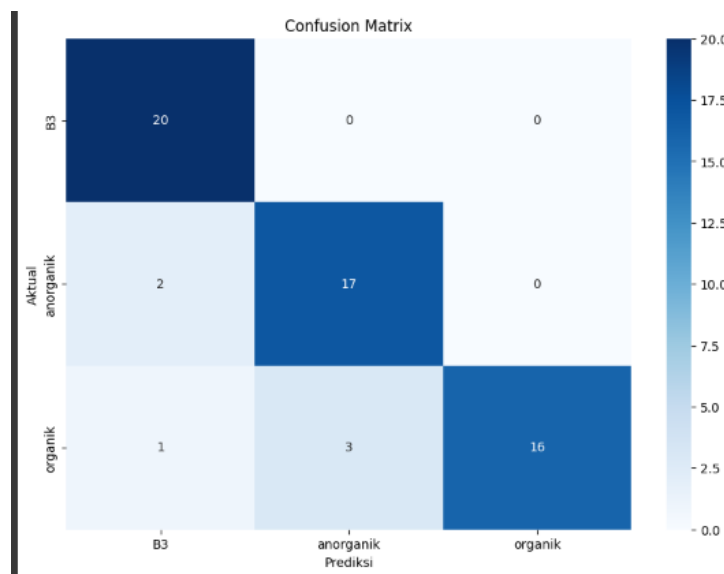
$$\text{TP} = 17$$

$$\text{Tinstance}(\text{Total sample}) = 20$$

$$\text{Accuracy} = 17 / 20$$

$$= 0,85 (85\%)$$

4.10 Evaluasi



Gambar 4.24 *Confusion Matrix ResNet-18*

Gambar 2.24 merupakan laporan klasifikasi yang ditampilkan untuk mengilustrasikan kinerja model pembelajaran mesin dalam mengklasifikasikan tiga kelas pada Jadi, jumlah total data yang diuji adalah 59 instance :

Table 1. 2 Perhitungan Nilai Akurasi Algoritma *ResNet18*

	TP/TN/FP/FN	$\text{Precision} = \frac{TP}{TP + FP}$	$\text{Recall} = \frac{TP}{TP + FN}$	$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$
Organik	16/39/0/4	1.0	0.8	0.88
Anorganik	17/37/3/2	0.85	0.89	0.87
B3	20/36/3/0	0,87	1.0	0.94
Rata Rata	-	0.90	0.90	0.89

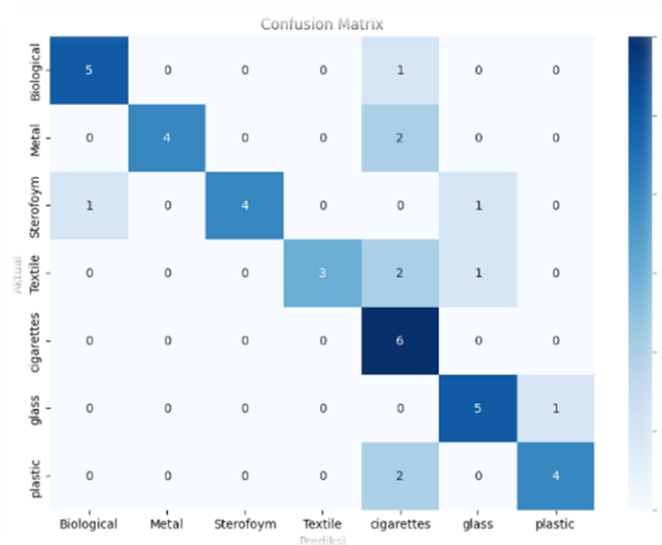
Total Prediksi Benar=20+17+16=54

$$\text{Accuracy} = \frac{\text{Total Prediksi Benar}}{\text{Total Instance}} = \frac{54}{60}$$

$$\text{Accuracy} = \frac{54}{60} = 0.9$$

Pada Table 1.2 terlihat Untuk kelas B3, model memiliki precision sebesar 0.87, recall 1.00, dan F1-score 0.93 dengan support 20, menunjukkan bahwa model ini sangat efektif dalam mengidentifikasi semua contoh kelas B3 dengan tepat. Kelas

anorganik memiliki precision 0.85, recall 0.89, dan F1-score 0.87 dengan support 19, menandakan performa yang baik meski tidak setinggi kelas B3. Sementara itu, kelas organik menunjukkan precision yang sempurna sebesar 1.00, namun recall-nya 0.80 dan F1-score 0.89 dengan support 20, yang berarti semua prediksi kelas organik benar, namun ada beberapa contoh kelas organik yang terlewatkan. Secara keseluruhan, model mencapai akurasi 0.91. Nilai rata-rata macro untuk precision, recall, dan F1-score adalah masing-masing 0.91, 0.90, dan 0.90, sementara nilai rata-rata tertimbang menunjukkan angka yang sama, yaitu 0.91, 0.90, dan 0.90. Hal ini menunjukkan bahwa model memiliki performa yang konsisten dan baik dalam mengklasifikasikan data dari ketiga kelas tersebut.



Gambar 5. 1 *Confusion Matrix ResNet18*

Gambar 5.1 merupakan hasil matrix kebingungan pada *Resnet18* Dilakukan 42

Instance dengan 7 Class yang berbeda dan didapat kesimpulan antara lain :

Table 1. 3 Menghitung Nilai akurasi Algoritma SSD

	TP/FP/FN/TN	$\text{Precision} = \frac{TP}{TP + FP}$	$\text{Recall} = \frac{TP}{TP + FN}$	$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$
Biological	5/0/1/36	1.0	0.83	0.9
Metal	4/0/2/36	1.0	0.66	0.8
Sterofyom	4/1/2/35	0.8	0.66	0.72
Textile	2/2/3/35	0.5	0.4	0.44
Cigarettes	6/0/0/36	1.0	1.0	1.0
Glass	5/1/1/35	0.83	0.83	0.83
Plastic	4/0/2/36	1.0	0.66	0.8
	Rata – rata	0.87	0.705	0.78

Total prediksi benar (jumlah elemen diagonal): 5+4+4+2+6+5+4=30

$$\text{Accuracy} = \frac{\text{Total Prediksi Benar}}{\text{Total Instance}} = \frac{30}{42} \approx 0.7143 (71.43\%)$$

Pada Table 1.3 berdasarkan perhitungan rata-rata metrik evaluasi, performa model klasifikasi sampah dapat dinilai cukup baik. Rata-rata akurasi model sebesar 94.61% menunjukkan bahwa hampir 95 dari 100 prediksi yang dibuat oleh model adalah benar. Ini berarti model sangat andal dalam mengklasifikasikan sampah secara keseluruhan. Rata-rata precision sebesar 87.6% menunjukkan bahwa ketika model memprediksi suatu jenis sampah sebagai kategori tertentu (misalnya, metal atau plastik), 87.6% dari waktu tersebut prediksinya benar. Ini penting untuk meminimalkan kesalahan ketika model mengidentifikasi sampah tertentu yang sebenarnya bukan.

Namun, rata-rata *recall* sebesar 70.57% mengindikasikan bahwa model mampu mengidentifikasi sekitar 70.57% dari semua sampah yang benar-benar termasuk dalam kategori positif, menunjukkan bahwa model mungkin melewatkan beberapa sampah yang seharusnya terklasifikasi (*false negatives*). *F1-Score* rata-rata sebesar 78.43% menunjukkan keseimbangan yang cukup baik antara precision dan recall, mengindikasikan bahwa model memiliki performa yang baik dalam hal mengidentifikasi dan mengklasifikasikan sampah dengan benar. Secara keseluruhan, metrik-metrik ini menunjukkan bahwa model klasifikasi sampah bekerja dengan baik, terutama dalam hal akurasi dan *precision*. Meskipun demikian, *recall* yang sedikit lebih rendah menunjukkan bahwa ada ruang untuk peningkatan, khususnya dalam kemampuan model untuk tidak melewatkan sampah yang seharusnya teridentifikasi. Penyesuaian lebih lanjut mungkin diperlukan untuk meningkatkan *recall* tanpa mengorbankan *precision*.

BAB 5

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berikut adalah kesimpulan berdasarkan analisis, perancangan, implementasi dan hasil pengujian sistem yang telah dilakukan :

1. Perhitungan akurasi saat mengidentifikasi objek sangat dipengaruhi oleh kondisi citra saat diuji. Akurasi tertinggi dicapai ketika citra dalam kondisi terang dan posisinya tepat tanpa halangan objek lain, sedangkan akurasi *miscprediction* disebabkan oleh kualitas gambar dan objek yang teramati di dalamnya seperti adanya objek sampah lain dan kualitas gambar . Kualitas dataset juga berpengaruh terhadap akurasi yang diperoleh saat menggunakan kedua algoritma. Semakin baik kualitas dataset, semakin tinggi nilai akurasi yang dihasilkan.
2. Berdasarkan Data yang di peroleh pada uji output terhadap 20 sample diluar dataset di peroleh hasil Implementasi Algoritma *ResNet-18* dan *SSD* dengan jumlah persentase 85% hal ini membuktikan bahwa akurasi sangat optimal walaupun dengan faktor tertentu yang mempengaruhi seperti dataset yang belum seimbang , kekurangan data dan variasi dataset
3. Pada model *SSD* nilai akurasi menghasilkan 75% Total dengan. *Precision* rata-rata sebesar 87.6% menunjukkan keandalan model dalam memprediksi jenis sampah tertentu dengan benar, sementara *recall* rata-rata sebesar 70.57% mengindikasikan bahwa model masih melewatkan beberapa sampah yang seharusnya terklasifikasi. *F1-Score* rata-rata sebesar 78.43% menunjukkan keseimbangan antara *precision* dan *recall*, menandakan performa yang baik.sedangkan pada *ResNet18* Menghasilkan nilai akurasi 81% Nilai rata-rata macro untuk *precision*, *recall*, dan *F1-score* adalah masing-masing 0.91, 0.90, dan 0.90, sementara nilai rata-rata tertimbang menunjukkan angka yang sama, yaitu 0.91, 0.90, dan 0.90. Hal ini menunjukkan bahwa model memiliki performa yang konsisten dan baik

5.2 Saran

Berikut merupakan saran yang dapat dipertimbangkan dalam pengembangan sistem selanjutnya:

1. Gunakan teknik augmentasi data seperti rotasi, flip, perubahan kecerahan, dan kontras untuk meningkatkan variasi dataset. Ini akan membantu model menjadi lebih robust terhadap variasi dalam data uji.
2. cobalah berbagai nilai *batch size* dan *learning rate* untuk menemukan kombinasi yang optimal. *Hyperparameter* tuning dapat secara signifikan meningkatkan akurasi model. valuasi jumlah *epoch* yang digunakan. Kadang-kadang, menambah jumlah epoch dapat membantu model untuk belajar lebih baik, tetapi juga perlu diwaspadai agar tidak terjadi *overfitting*
3. Pastikan dataset yang digunakan seimbang dan menambah varian dataset, mencakup berbagai kondisi pencahayaan, sudut pandang, dan jenis halangan. Dataset yang tidak seimbang dapat menyebabkan model *overfitting* pada kelas yang dominan. Pastikan objek yang akan diidentifikasi memiliki posisi dan sudut pandang yang konsisten. Variasi yang besar dalam posisi dan sudut pandang dapat mempersulit model untuk mengidentifikasi objek dengan akurat.

DAFTAR PUSTAKA

- Abdurahman, Rasidi; Pasaribu, Yolanda Al hidayah; Ziqri, Afzal; Adiharma, Faisal Adhinanta;. (2022). Klasifikasi Sampah Organik dan Non-Organik Menggunakan Convolutional Neural Network. *Jurnal Teknik Informatika dan Sistem Informasi*, 142-143.
- Aningtiyas, P. R., Sumin, A., & Wirawan, S. (2020). Pembuatan Aplikasi Deteksi Objek Menggunakan TensorFlow Object Detection API dengan Memanfaatkan SSD MobileNet V2 Sebagai Model Pra-Terlatih. *Jurnal Ilmiah KOMPUTASI*, 424.
- Aviya, W. M., Irfansyah, A., & Puspita, R. D. (2020). FACE DETECTION MENGGUNAKAN METODE ARTIFICIAL NEURAL NETWORK(ANN) BERBASIS RASPBERRY PI SEBAGAI IDENTIFIKASI EKSPRESI WAJAH. *Politeknik Penerbangan Surabaya*, 1-2.
<https://doi.org/10.46491/snitp.v4i1.787>
- Liu , W., Anguelov, D., Erhan, D., Szegedy, C., & Reed, S. (2019). *SSD Singel Shot Multibox Detector*. USA: Springer International Publishing AG.
https://link.springer.com/chapter/10.1007/978-3-319-46448-0_2
- Nugroho, P. A., Fenriana, I., & Arijanto, R. (2020). IMPLEMENTASI DEEP LEARNING MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK (CNN)PADA EKSPRESI MANUSIA. *JURNAL ALGOR*, 13.
- Pernando, Muttaqin, M. R., & Ramadhan, Y. R. (2023). DETEKSI JENIS SAMPAH SECARA REALTIMEMENGGUNAKAN METODE SINGLE SHOT MULTIBOX DETECTOR (SSD). *Jurnal Mahasiswa Teknik Informatika*.
<https://doi.org/10.36040/jati.v7i3.6976>
- Prasmatio, M., Rahmat, B., & Yuniar, I. (2020). DETEKSI DAN PENGENALAN IKAN MENGGUNAKAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK. *Jurnal Informatika dan Sistem Informasi (JIFoSI)*.
<http://download.garuda.kemdikbud.go.id/>
- Stephen, Raymond, & Santoso, H. (2019). APLIKASI CONVOLUTION NEURAL NETWORK UNTUK MENDETEKSI JENIS-JENIS SAMPAH. *Jurnal Sistem Informasi dan Telematika ISSN 2087-2062/E-ISSN 2686-181X*, 124.
- Sunanto, O. D., & Utomo, P. H. (2022). IMPLEMENTASI DEEP LEARNING DENGAN CONVOLUTIONAL NEURAL NETWORK UNTUK KLASIFIKASI GAMBAR SAMPAH ORGANIK DAN ANORGANIK. *Seminar Nasional Matematika, Geometri, Statistika, dan KomputasiSeNa-MaGeStiK*.
<https://jurnal.unej.ac.id/index.php/prosiding/article/view/33527/11688>
- Syahputra, Z. (2023). Penerapan SSD-MobileNetDalamIdentifikasiJenis Buah Apel. *INDOTECH Indonesian Journal of Education AndComputer Science*, 2.
<https://doi.org/10.60076/indotech.v1i1.2>

Zayd, M., Oktavian, M., T. Meranggi, D., Figo, J., & Yudistira, N. (2022). Perbaikan klasifikasi sampah menggunakan pretrained Convolutional Neural Network. *Jurnal Ilmiah Sistem Informasi*.

LAMPIRAN

Lampiran 1. *Listing Program*

```
# flask init
app = Flask(__name__)
app.config['UPLOAD_FOLDER'] = 'static/uploads/'
app.config['RESULT_FOLDER'] = 'static/result/'

# classify using ResNet
def predict_image(img, model, device):
    transform = tt.Compose([
        tt.Resize((128, 128)),
        tt.ToTensor()
    ])
    img = transform(img).unsqueeze(0).to(device)
    model.eval()
    with torch.no_grad():
        yb = model(img)
        conf, preds = torch.max(yb, dim=1)
    return preds[0].item(), conf[0].item()

class IntelImageResnet(nn.Module):
    def __init__(self):
        super(IntelImageResnet, self).__init__()
        self.network = models.resnet18(pretrained=True)
        num_fters = self.network.fc.in_features
        self.network.fc = nn.Linear(num_fters, 6)

    def forward(self, xb):
        return torch.sigmoid(self.network(xb))

def get_default_device():
    if torch.cuda.is_available():
        return torch.device('cuda')
    else:
        return torch.device('cpu')

device = get_default_device()

model_path = 'my-cource-proj-ffn-resnet50_v4.pth'
model = IntelImageResnet()
model.load_state_dict(torch.load(model_path, map_location=device))
model = model.to(device)

# test_dataset_classes = ['B3', 'anorganik', 'organik']
test_dataset_classes = ['B3', 'anorganik', 'organik']
# end classify using ResNet

# classify using SSD
rf = roboflow.Roboflow(api_key="DsZVNB7AJs4VV3QqsDyP")
project = rf.workspace().project("ssd-xyvcj")
model2 = project.version("1").model
model2.confidence = 20
model2.overlap = 10
```



```

def draw_prediction(image_path, prediction):
    image = Image.open(image_path)
    draw = ImageDraw.Draw(image)

    x = prediction['x']
    y = prediction['y']
    w = prediction['width']
    h = prediction['height']
    confidence = prediction['confidence']
    label = prediction['class']

    left = x - w / 2
    top = y - h / 2
    right = x + w / 2
    bottom = y + h / 2

    draw.rectangle([left, top, right, bottom], outline='red', width=3)
    draw.text((left, top - 10), f'{label}: {confidence:.2f}', fill='red')

    output_path = os.path.join(app.config['RESULT_FOLDER'], 'output.png')
    image.save(output_path)
    return output_path
# end classify using SSD

@app.route('/', methods=['GET'])
def index():
    return render_template('index.html')

@app.route('/upload', methods=['GET', 'POST'])
def upload():
    if request.method == 'POST':
        file = request.files['file']
        if file:
            filepath = os.path.join(app.config['UPLOAD_FOLDER'],
file.filename)
            file.save(filepath)

            # prediction ResNet
            img = Image.open(filepath).convert('RGB')
            predictionResNetResult, confidenceResNetResult =
predict_image(img, model, device)

            # prediction SSD
            predictionSSD = model2.predict(filepath)
            predictionSSDF = predictionSSD.json()['predictions']

            if predictionSSDF:
                firstPredictionSSD = predictionSSDF[0]

                output_path = draw_prediction(filepath,
firstPredictionSSD)

                return render_template('upload.html', image_url=output_path,
predictionSSDResult=firstPredictionSSD,
predictionResNetResult=test_dataset_classes[predictionResNetResult],
confidenceResNetResult=confidenceResNetResult)
            # return render_template('upload.html',
predictionResNetResult=test_dataset_classes[predictionResNetResult],
confidenceResNetResult=confidenceResNetResult)
            return render_template('upload.html', image_url=None,

```

```

predictionSSDResult=None, predictionResNetResult=None)

if __name__ == '__main__':
    app.run(debug=True)

Upload .HTML

<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">
    <title>Image Classification</title>
    <link
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.c
ss" rel="stylesheet">
    <style>
      body {
        font-family: Arial, sans-serif;
        margin: 0;
        padding: 0;
        background-color: #ec135494;
      }

      .container {
        max-width: 800px;
        margin: 50px auto;
        padding: 20px;
        border: 1px solid #ddd;
        border-radius: 5px;
        background-color: #d5d3dd93;
        box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
      }

      h1 {
        text-align: center;
        margin-bottom: 20px;
      }

      form {
        display: flex;
        justify-content: center;
        margin-bottom: 40px;
        background-color: rgba(167, 179, 0, 0.534);
      }

      input[type="file"] {
        padding: 10px;
        border: 1px solid #eee6e6;
        border-radius: 5px;
        border-color: #ccc;
        cursor: pointer;
      }

      button[type="submit"] {
        padding: 10px 20px;
        background-color: #ad13bbb4;

```

```

        color: #fff;
        border: none;
        border-radius: 5px;
        cursor: pointer;
    }

    .results {
        margin-top: 30px;
        justify-content: space-between;
        align-items: center;
    }

    .prediction-image {
        width: 200px;
        display: block;
    }

    .predictions {
        list-style: none;
        padding: 0;
        margin: 20px;
    }

    .predictions li {
        margin-bottom: 10px;
        font-weight: bold;
    }

</style>
</head>
<body>

    <div class="container">
        <h1>Upload an Image for Classification</h1>
        <form method="post" enctype="multipart/form-data">
            <input type="file" name="file" required>
            <button type="submit">Upload</button>
        </form>
        {% if image_url %}
        <div class="results">
            <h2>Prediction SSD</h2>
            
            <ul>
                <li>{{ predictionSSDResult.class }} </li>
                <!-- {{ predictionSSDResult.confidence }} -->
            </ul>
            <h2>Prediction ResNet</h2>
            <ul>
                <li>{{ predictionResNetResult }} </li>
                <!-- : {{ confidenceResNetResult }} -->
            </ul>
        </div>
        {% endif %}
    </div>

    <!-- Link to Bootstrap JS and dependencies -->
    <script src="https://code.jquery.com/jquery-
3.5.1.slim.min.js"></script>
    <script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.5.3/dist/umd/popper.mi

```

```
n.js"></script>
  <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"
></script>
  </body>
</html>
```