BAB 2
LANDASAN TEORI

Landasan teori membahas konsep yang mendukung klasifikasi *multi-label* sampah anorganik menggunakan EfficientNet-B0 dengan prapemrosesan segmentasi semantik multikelas U-Net. Pembahasan meliputi penelitian terdahulu, klasifikasi *multi-label*, segmentasi citra, CNN, *transfer learning*, EfficientNet-B0, U-Net, serta evaluasi model.

2.1 Penelitian Terdahulu

Penelitian mengenai klasifikasi sampah berbasis citra telah banyak dikembangkan menggunakan pendekatan *deep learning*, *transfer learning*, dan pengembangan arsitektur CNN. Beberapa penelitian berfokus pada peningkatan performa model klasifikasi, sedangkan penelitian lain memanfaatkan arsitektur segmentasi seperti U-Net untuk memisahkan area objek dan latar belakang pada berbagai domain citra. Ringkasan penelitian terdahulu yang relevan dengan penelitian ini ditunjukkan pada Tabel 2.1.

Tabel 2.1. Penelitian Terdahulu

No.	Penelitian	Metode	Hasil dan Relevansi
1	Yong, Ma, Sun, dan Du [6]	MobileNetV2 berbasis *transfer learning* untuk klasifikasi citra sampah.	Menunjukkan bahwa model ringan berbasis CNN dapat digunakan untuk klasifikasi sampah dan menghasilkan performa lebih baik dibandingkan CNN sederhana.
2	Poudel dan Poudyal [19]	CNN berbasis *transfer learning* dengan beberapa model pra-latih.	Mendukung penggunaan model pra-latih dalam klasifikasi material sampah berbasis citra.

Tabel 2.1. Penelitian Terdahulu (lanjutan)

No.	Penelitian	Metode	Hasil dan Relevansi
3	Shetty, Kallianpur, Fernandes, Rodrigues, dan Padmanabha [9]	*Transfer learning*, model *hybrid*, dan *ensemble* CNN.	Menunjukkan bahwa kombinasi beberapa model CNN dapat meningkatkan performa klasifikasi sampah.
4	Risfendra, Ananda, dan Setyawan [7]	EfficientNet-B0 berbasis *transfer learning*.	Relevan karena menggunakan EfficientNet-B0 untuk klasifikasi sampah dan memperoleh akurasi uji sebesar 91,94%.
5	Ayman, Rahman, Rahim, Saha, Haque, dan Rahat [8]	Perbandingan EfficientNet-B0, EfficientNetB2, EfficientNetB3, dan EfficientNetB4.	Menunjukkan bahwa varian EfficientNet dapat digunakan secara efektif untuk klasifikasi citra sampah, dengan EfficientNet-B0 dan EfficientNetB2 mencapai akurasi 93%.
6	Alshanab [20]	U-Net untuk segmentasi semantik pada citra lingkungan urban.	Menunjukkan bahwa U-Net dapat digunakan untuk memisahkan objek dan latar belakang pada citra dengan struktur visual yang beragam.
7	Zhang, Jiang, Zheng, dan Yao [17]	Improved U-Net berbasis *transfer learning* untuk segmentasi citra *remote sensing*.	Mendukung penggunaan U-Net sebagai model segmentasi yang dapat menghasilkan area objek yang lebih terfokus.
8	Wu dkk. [10]	Query2Label dengan ViT-B/16 untuk klasifikasi *multi-label* sampah perkotaan.	Menunjukkan bahwa satu citra sampah dapat memuat beberapa kelas dan perlu diprediksi menggunakan pendekatan *multi-label*.

Tabel 2.1. Penelitian Terdahulu (lanjutan)

No.	Penelitian	Metode	Hasil dan Relevansi
9	Umar, Asrul, dan Yuyun [11]	Perbandingan beberapa arsitektur CNN, termasuk EfficientNet-B0, untuk klasifikasi *multi-label* sampah.	Mendukung penggunaan EfficientNet-B0 sebagai salah satu model pembanding pada klasifikasi citra dengan beberapa objek sampah.
10	Zhang dkk. [18]	WISNet untuk menghasilkan *pseudo-label* segmentasi semantik pada citra sampah yang padat dan tidak seimbang.	Menunjukkan bahwa beberapa kategori sampah dalam satu citra dapat direpresentasikan melalui segmentasi pada tingkat piksel.
11	Shihab dkk. [21]	Segmentasi U-Net, pemilihan kontur, *cropping*, dan klasifikasi menggunakan Vision Transformer.	Mendukung penggunaan hasil segmentasi sebagai tahap prapemrosesan sebelum klasifikasi.

Penelitian terdahulu menunjukkan bahwa klasifikasi *multi-label* dapat digunakan untuk mengenali beberapa jenis sampah dalam satu citra, sedangkan segmentasi semantik digunakan untuk memetakan kelas objek pada tingkat piksel. Hasil segmentasi juga dapat dimanfaatkan untuk membentuk area objek sebelum tahap klasifikasi. Namun, kajian yang mengevaluasi segmentasi semantik multikelas U-Net sebagai prapemrosesan EfficientNet-B0 pada klasifikasi *multi-label* sampah anorganik berbasis data primer masih terbatas. Oleh karena itu, penelitian ini membandingkan klasifikasi citra asli dan citra hasil prapemrosesan segmentasi.

2.2 Sampah Anorganik dalam Klasifikasi Citra

Penelitian ini membatasi sampah anorganik menjadi lima kelas, yaitu botol kaca, botol plastik kemasan tebal, botol plastik tipis, kaleng, dan kemasan plastik. Setiap kelas memiliki karakteristik visual yang berbeda, seperti bentuk, warna, tekstur, transparansi, ketebalan material, dan kondisi fisik objek. Karakteristik

tersebut menjadi informasi yang dipelajari model untuk membedakan setiap kelas [22, 23].

Setiap citra memuat dua objek dari dua kelas berbeda sehingga memiliki dua label aktif. Keberadaan lebih dari satu kelas meningkatkan kompleksitas klasifikasi karena objek dapat memiliki ukuran, posisi, orientasi, dan tingkat tumpang tindih yang berbeda. Model perlu mengenali kedua kelas tanpa mengabaikan objek yang memiliki area visual lebih kecil.

2.3 Klasifikasi Citra *Multi-Label* Berbasis *Deep Learning*

Klasifikasi citra berbasis *deep learning* memanfaatkan jaringan saraf untuk mempelajari fitur visual secara otomatis dari data. Berbeda dengan metode konvensional yang memerlukan perancangan fitur secara manual, model *deep learning* dapat mempelajari pola langsung selama proses pelatihan [24]. Pada klasifikasi *multi-label*, satu citra dapat dipetakan ke dalam lebih dari satu label kelas. Pendekatan ini sesuai dengan penelitian karena setiap citra memuat dua objek dari dua kelas sampah yang berbeda.

Dataset klasifikasi *multi-label* dapat dinyatakan sebagai $\mathscr{D} = \{(x_i, \mathbf{y}_i)\}_{i=1}^{N}$, dengan x_i sebagai citra ke-i dan $\mathbf{y}_i$ sebagai vektor label. Vektor tersebut berisi nilai biner yang menunjukkan keberadaan setiap kelas pada citra.

$$\mathbf{y}_i \in \{0,1\}^K \tag{2.1}$$

Variabel K menunjukkan jumlah seluruh kelas dan pada penelitian ini bernilai 5. Elemen $y_{i,k}$ bernilai 1 apabila kelas ke-k terdapat pada citra ke-i dan bernilai 0 apabila kelas tersebut tidak terdapat pada citra. Karena setiap citra memuat dua objek dari dua kelas berbeda, setiap vektor label memiliki dua elemen aktif. Model tidak digunakan untuk menentukan lokasi atau jumlah objek, tetapi untuk memprediksi keberadaan kelas pada tingkat citra.

Model berbasis *Convolutional Neural Network* mempelajari representasi visual secara bertingkat, mulai dari fitur sederhana seperti tepi, warna, dan tekstur hingga fitur yang lebih kompleks seperti bentuk dan struktur objek [25]. Dalam klasifikasi *multi-label*, model perlu mengenali karakteristik lebih dari satu kelas dalam citra yang sama. Hal tersebut menjadi lebih menantang apabila objek memiliki ukuran, posisi, atau tingkat tumpang tindih yang berbeda. Oleh karena itu, keluaran setiap kelas dihitung secara independen.

Setiap nilai *logit* yang dihasilkan model dikonversi menjadi probabilitas menggunakan fungsi sigmoid. Berbeda dengan *softmax*, fungsi sigmoid tidak mengharuskan jumlah probabilitas seluruh kelas bernilai 1. Dengan demikian, lebih dari satu kelas dapat memiliki probabilitas tinggi secara bersamaan. Fungsi sigmoid dirumuskan sebagai berikut.

$$p_{i,k} = \sigma(z_{i,k}) = \frac{1}{1+e^{-z_{i,k}}} \tag{2.2}$$

Variabel $z_{i,k}$ menunjukkan nilai *logit* untuk citra ke-i pada kelas ke-k, sedangkan $p_{i,k}$ menunjukkan probabilitas keberadaan kelas tersebut. Probabilitas kemudian dikonversi menjadi label prediksi menggunakan nilai ambang atau *threshold* τ.

$$\hat{y}_{i,k} = \begin{cases} 1, & \text{jika } p_{i,k} \geq \tau, \\ 0, & \text{jika } p_{i,k} < \tau. \end{cases} \tag{2.3}$$

Variabel $\hat{y}_{i,k}$ menunjukkan hasil prediksi kelas ke-k pada citra ke-i. Nilai 1 menunjukkan bahwa kelas diprediksi terdapat pada citra, sedangkan nilai 0 menunjukkan bahwa kelas tidak diprediksi. Nilai *threshold* ditentukan menggunakan data validasi dan selanjutnya diterapkan pada data uji. Pendekatan ini memungkinkan EfficientNet-B0 menghasilkan lebih dari satu label prediksi pada satu citra.

2.4 Segmentasi Citra

Segmentasi citra menghasilkan prediksi pada tingkat piksel sehingga lokasi dan batas objek dapat direpresentasikan [26]. Penelitian ini menggunakan segmentasi semantik multikelas dengan enam kelas, yaitu satu kelas latar belakang dan lima kelas sampah anorganik. Setiap piksel hanya memperoleh satu label kelas dan objek dari kelas yang sama tidak dibedakan sebagai instans terpisah.

$$M(x,y) \in \{0,1,2,3,4,5\} \tag{2.4}$$

Nilai 0 menunjukkan latar belakang, sedangkan nilai 1 sampai 5 menunjukkan kelas objek. Untuk kebutuhan prapemrosesan klasifikasi, seluruh

kelas objek digabungkan menjadi satu area *foreground*.

$$M_f(x,y) = \begin{cases} 0, & M(x,y) = 0, \\ 1, & M(x,y) > 0. \end{cases} \tag{2.5}$$

Mask foreground digunakan untuk mempertahankan piksel RGB objek dan mengganti area latar belakang.

$$I'_c(x,y) = I_c(x,y)M_f(x,y) + B_c\left(1 - M_f(x,y)\right) \tag{2.6}$$

Proses *cropping* dilakukan berdasarkan batas minimum dan maksimum seluruh piksel *foreground*. Batas tersebut mencakup gabungan seluruh objek agar label yang terdapat pada citra tetap dipertahankan [27, 28].

2.5 *Convolutional Neural Network* (CNN)

Convolutional Neural Network merupakan arsitektur *deep learning* yang mampu mengekstraksi fitur visual secara otomatis melalui operasi konvolusi. Lapisan awal umumnya mempelajari pola sederhana seperti tepi dan tekstur, sedangkan lapisan yang lebih dalam mempelajari bentuk dan struktur objek. Lapisan *pooling* digunakan untuk mengurangi ukuran spasial representasi fitur [29].

Dalam penelitian ini, CNN menjadi dasar bagi EfficientNet-B0 dan U-Net. EfficientNet-B0 menggunakan fitur hasil konvolusi untuk menghasilkan prediksi *multi-label* pada tingkat citra, sedangkan U-Net menggunakannya untuk menghasilkan kelas pada tingkat piksel. Dengan demikian, kedua arsitektur menggunakan prinsip CNN untuk tugas yang berbeda.

2.6 Transfer Learning

Transfer learning memanfaatkan pengetahuan dari model yang telah dilatih pada domain sumber untuk menyelesaikan tugas pada domain target. Model *pre-trained* telah mempelajari pola visual umum sehingga pelatihan pada dataset target tidak dimulai dari parameter acak [30].

Pendekatan ini dapat dilakukan melalui *feature extraction* atau *fine-tuning*. Pada *feature extraction*, lapisan dasar digunakan sebagai ekstraktor fitur, sedangkan pada *fine-tuning* sebagian lapisan diperbarui untuk menyesuaikan karakteristik data

target. Dalam penelitian ini, EfficientNet-B0 menggunakan bobot awal ImageNet dan lapisan keluarannya disesuaikan untuk klasifikasi *multi-label* lima kelas.

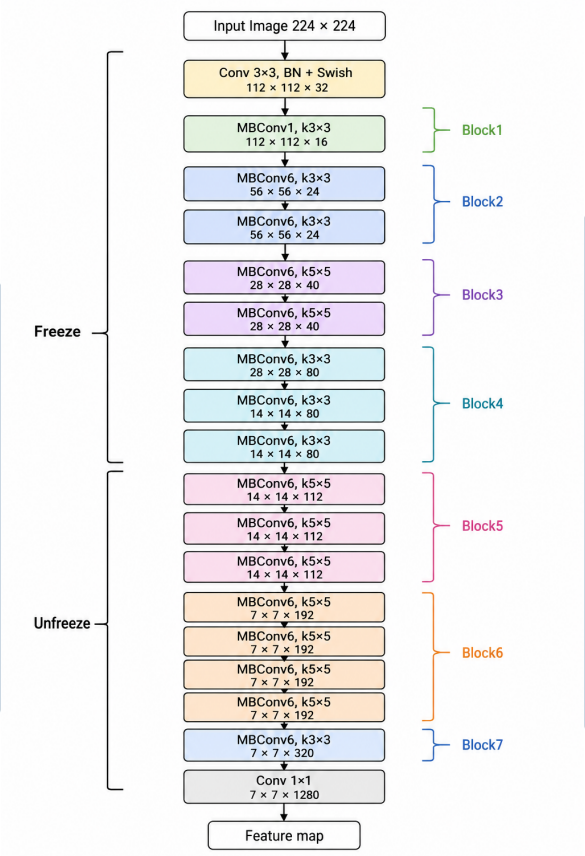
2.7 EfficientNet-B0

EfficientNet-B0 merupakan salah satu arsitektur *Convolutional Neural Network* dari keluarga EfficientNet yang dirancang untuk memperoleh keseimbangan antara kemampuan ekstraksi fitur, akurasi, dan efisiensi komputasi. EfficientNet dikembangkan menggunakan pendekatan *compound scaling*, yaitu strategi penskalaan yang menyesuaikan kedalaman jaringan, lebar jaringan, dan resolusi citra masukan secara seimbang [12]. EfficientNet-B0 merupakan model dasar dari keluarga EfficientNet, sedangkan varian yang lebih besar dikembangkan melalui penskalaan terhadap model dasar tersebut.

2.7.1 Arsitektur EfficientNet-B0

Arsitektur EfficientNet-B0 terdiri dari lapisan konvolusi awal, rangkaian blok *Mobile Inverted Bottleneck Convolution* atau MBConv, dan lapisan akhir untuk menghasilkan representasi fitur citra. Lapisan konvolusi awal digunakan untuk mengekstraksi fitur dasar dari citra masukan, sedangkan blok MBConv digunakan untuk mempelajari representasi fitur yang lebih kompleks secara bertahap.

Selama proses ekstraksi fitur, ukuran spasial *feature map* berkurang secara bertahap, sedangkan jumlah kanal fitur meningkat. Mekanisme tersebut memungkinkan model mempelajari fitur visual mulai dari pola sederhana, seperti tepi, warna, dan tekstur, hingga pola yang lebih kompleks, seperti bentuk dan struktur objek. Pada bagian akhir, representasi fitur yang dihasilkan digunakan sebagai masukan bagi lapisan klasifikasi.



Gambar 2.1. Ilustrasi arsitektur EfficientNet-B0 yang diadaptasi dari konsep EfficientNet oleh Tan dan Le [12].

Gambar 2.1 menunjukkan alur umum arsitektur EfficientNet-B0. Citra masukan berukuran 224×224 piksel diproses melalui lapisan konvolusi awal dan rangkaian blok MBConv. Fitur yang dihasilkan kemudian diproses pada bagian akhir jaringan untuk membentuk representasi citra sebelum diteruskan ke lapisan klasifikasi. Lapisan klasifikasi akhir dapat disesuaikan dengan jumlah kelas dan jenis tugas pada dataset target.

2.7.2 *Compound Scaling*

Salah satu konsep utama pada keluarga EfficientNet adalah *compound scaling*. Pendekatan ini meningkatkan kapasitas model dengan menyesuaikan tiga aspek utama secara bersamaan, yaitu kedalaman jaringan, lebar jaringan, dan resolusi citra masukan [12]. Kedalaman jaringan berkaitan dengan jumlah lapisan, lebar jaringan berkaitan dengan jumlah kanal atau *filter*, sedangkan resolusi berkaitan dengan ukuran citra yang diproses oleh model.

Berbeda dengan pendekatan penskalaan yang hanya memperbesar salah satu aspek jaringan, *compound scaling* mengatur ketiganya secara seimbang. Secara matematis, prinsip tersebut dinyatakan melalui koefisien kedalaman d, lebar w, dan resolusi r. Jika ϕ merupakan koefisien gabungan, maka penskalaan model dapat dirumuskan sebagai berikut.

$$d = \alpha^{\phi}, \qquad w = \beta^{\phi}, \qquad r = \gamma^{\phi}, \qquad \alpha\beta^2\gamma^2 \approx 2 \tag{2.7}$$

Variabel d, w, dan r masing-masing menunjukkan penskalaan kedalaman, lebar, dan resolusi, sedangkan ϕ mengatur tingkat penskalaan model. Batasan tersebut menjaga peningkatan kapasitas dan kebutuhan komputasi tetap seimbang [12].

2.7.3 EfficientNet-B0 dalam Klasifikasi Sampah

EfficientNet-B0 memiliki ukuran yang relatif ringan dibandingkan varian EfficientNet yang lebih besar, tetapi tetap mampu mengekstraksi fitur visual secara efektif. Karakteristik tersebut relevan untuk klasifikasi sampah anorganik karena setiap kelas memiliki ciri visual yang berbeda, seperti transparansi dan pantulan cahaya pada botol kaca, tekstur logam pada kaleng, perbedaan ketebalan pada botol plastik, serta bentuk fleksibel pada kemasan plastik.

Penggunaan EfficientNet-B0 pada klasifikasi sampah telah dibahas dalam beberapa penelitian sebelumnya. EfficientNet-B0 berbasis *transfer learning* dapat digunakan untuk klasifikasi sampah berbasis citra dengan performa yang kompetitif [7]. Penelitian lain juga menunjukkan bahwa EfficientNet-B0 mampu memberikan performa yang baik dibandingkan beberapa varian EfficientNet lainnya pada tugas klasifikasi citra sampah [8]. Selain itu, EfficientNet-B0 juga telah digunakan sebagai salah satu arsitektur pada klasifikasi *multi-label* sampah yang memuat beberapa objek dalam satu citra [11].

Dalam penelitian ini, EfficientNet-B0 digunakan sebagai model klasifikasi *multi-label* untuk mengenali lima kelas sampah anorganik, yaitu botol kaca, botol plastik kemasan tebal, botol plastik tipis, kaleng, dan kemasan plastik. Lapisan klasifikasi akhir disesuaikan menjadi lima neuron keluaran dengan fungsi aktivasi sigmoid. Setiap neuron menghasilkan probabilitas keberadaan satu kelas secara independen sehingga dua kelas dapat diprediksi secara bersamaan pada satu citra.

EfficientNet-B0 diterapkan pada dua skenario eksperimen. Skenario

pertama menggunakan citra asli tanpa segmentasi, sedangkan skenario kedua menggunakan citra RGB hasil *masking* dan *cropping* berdasarkan prediksi segmentasi semantik multikelas U-Net. Arsitektur, konfigurasi pelatihan, pembagian data, dan metrik evaluasi EfficientNet-B0 dibuat sama pada kedua skenario. Dengan demikian, perbedaan kinerja yang diperoleh dapat digunakan untuk mengevaluasi pengaruh prapemrosesan segmentasi terhadap klasifikasi *multi-label*.

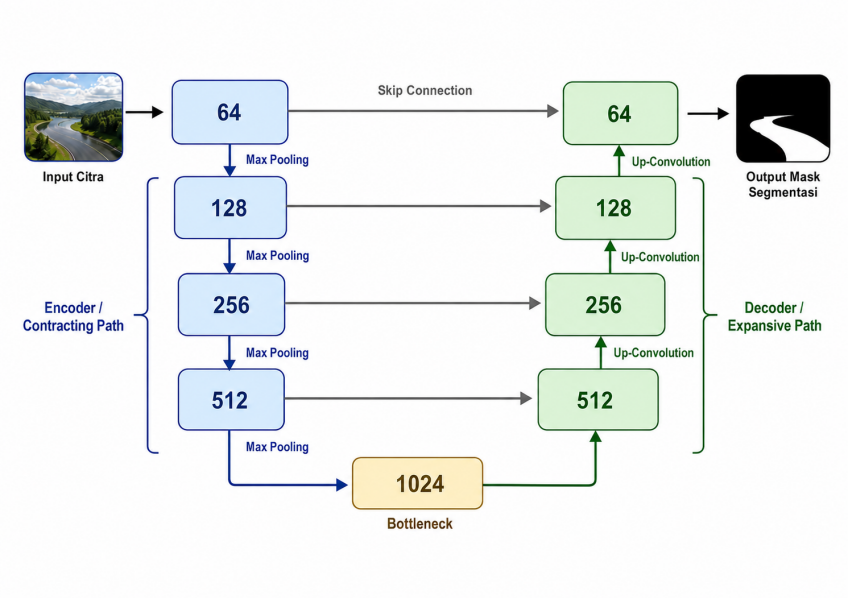
2.8 U-Net

U-Net merupakan arsitektur *deep learning* yang banyak digunakan untuk segmentasi citra karena memiliki struktur *encoder-decoder* dan *skip connection*. Berbeda dengan model klasifikasi yang menghasilkan prediksi pada tingkat citra, U-Net menghasilkan prediksi pada tingkat piksel sehingga lokasi, bentuk, dan batas area objek dapat diketahui [15]. Dalam penelitian ini, U-Net digunakan untuk melakukan segmentasi semantik multikelas terhadap lima kelas sampah anorganik dan satu kelas latar belakang. Hasil segmentasi tersebut selanjutnya digunakan sebagai dasar proses *masking* dan *cropping* sebelum citra diklasifikasikan menggunakan EfficientNet-B0.

2.8.1 Arsitektur U-Net

Arsitektur U-Net terdiri atas bagian *encoder*, *bottleneck*, dan *decoder*. Bagian *encoder* berfungsi mengekstraksi fitur citra melalui rangkaian operasi konvolusi dan *downsampling*. Proses tersebut mengurangi ukuran spasial *feature map* secara bertahap sekaligus meningkatkan kemampuan model dalam mempelajari fitur semantik. Bagian *bottleneck* menjadi penghubung antara *encoder* dan *decoder* serta menyimpan representasi fitur dengan tingkat abstraksi yang lebih tinggi.

Bagian *decoder* kemudian meningkatkan kembali resolusi spasial melalui proses *upsampling*. Fitur dari bagian *decoder* digabungkan dengan fitur dari bagian *encoder* melalui *skip connection*. Penggabungan tersebut membantu model mempertahankan informasi lokasi dan batas objek yang dapat berkurang selama proses *downsampling*. Pada bagian akhir, U-Net menghasilkan peta probabilitas kelas untuk setiap piksel pada citra.



Gambar 2.2. Ilustrasi arsitektur U-Net untuk segmentasi citra yang diadaptasi dari konsep arsitektur U-Net oleh Ronneberger, Fischer, dan Brox [15].

Gambar 2.2 menunjukkan alur umum arsitektur U-Net. Bagian *encoder* menurunkan resolusi citra untuk memperoleh fitur semantik, sedangkan bagian *decoder* meningkatkan kembali resolusi untuk membentuk keluaran segmentasi. Dalam penelitian ini, keluaran U-Net berupa *mask* segmentasi semantik multikelas yang membedakan kelas latar belakang dan lima kelas sampah anorganik.

2.8.2 *Skip Connection*

Salah satu komponen utama pada arsitektur U-Net adalah *skip connection*, yaitu hubungan antara *feature map* pada bagian *encoder* dan *decoder* yang memiliki tingkat resolusi sama. Mekanisme ini membantu mempertahankan informasi spasial yang dapat berkurang selama proses *downsampling*. Dengan adanya *skip connection*, informasi mengenai lokasi, bentuk, dan batas objek dari bagian *encoder* dapat digunakan kembali pada proses rekonstruksi di bagian *decoder*.

Jika fitur dari *encoder* pada tingkat ke-l dinyatakan sebagai E_l dan keluaran *upsampling* dari *decoder* pada tingkat yang sama dinyatakan sebagai D_l, proses penggabungan fitur dapat dituliskan sebagai berikut.

$$F_l = \text{Concat}(E_l, D_l) \tag{2.8}$$

Variabel F_l merupakan hasil penggabungan fitur yang digunakan sebagai masukan pada bagian *decoder*. Variabel E_l menunjukkan *feature map* dari bagian *encoder* yang membawa informasi spasial, sedangkan D_l merupakan fitur hasil *upsampling* pada bagian *decoder*. Fungsi Concat digunakan untuk menggabungkan kedua fitur tersebut pada dimensi kanal.

2.8.3 Segmentasi Semantik Multikelas pada U-Net

Segmentasi semantik multikelas merupakan proses pemberian satu label kelas pada setiap piksel citra. Dalam penelitian ini, U-Net menghasilkan enam kelas keluaran, yaitu satu kelas latar belakang dan lima kelas sampah anorganik. Lima kelas objek tersebut terdiri dari botol kaca, botol plastik kemasan tebal, botol plastik tipis, kaleng, dan kemasan plastik.

Jika jumlah seluruh kelas segmentasi dinyatakan sebagai C, maka pada penelitian ini berlaku $C = 6$. Untuk setiap piksel pada koordinat (x,y), U-Net menghasilkan nilai *logit* $z_k(x,y)$ untuk setiap kelas ke-k. Nilai tersebut kemudian dikonversi menjadi probabilitas menggunakan fungsi *softmax* sebagai berikut.

$$P_k(x,y) = \frac{e^{z_k(x,y)}}{\sum_{j=0}^{C-1} e^{z_j(x,y)}} \tag{2.9}$$

Variabel $P_k(x,y)$ menunjukkan probabilitas bahwa piksel pada koordinat (x,y) termasuk ke dalam kelas ke-k. Variabel $z_k(x,y)$ merupakan nilai *logit* untuk kelas tersebut, sedangkan C menyatakan jumlah seluruh kelas segmentasi. Fungsi *softmax* menghasilkan distribusi probabilitas antarkelas pada setiap piksel sehingga jumlah probabilitas seluruh kelas untuk satu piksel bernilai 1.

Kelas prediksi pada setiap piksel diperoleh dengan memilih kelas yang memiliki probabilitas tertinggi.

$$\hat{M}(x,y) = \arg \max_{k \in \{0,1,\ldots,C-1\}} P_k(x,y) \tag{2.10}$$

Variabel $\hat{M}(x,y)$ menunjukkan kelas hasil prediksi pada koordinat piksel (x,y). Nilai 0 menunjukkan kelas latar belakang, sedangkan nilai 1 sampai 5 menunjukkan lima kelas sampah anorganik. Berbeda dengan segmentasi biner, penentuan kelas piksel pada segmentasi multikelas tidak dilakukan menggunakan satu nilai ambang, tetapi dengan memilih kelas yang memiliki probabilitas tertinggi.

Segmentasi semantik memberikan satu kelas pada setiap piksel, tetapi tidak membedakan dua objek dari kelas yang sama sebagai instans yang terpisah. Oleh karena itu, U-Net dalam penelitian ini tidak digunakan untuk menghitung jumlah objek maupun melakukan *instance segmentation*. U-Net digunakan untuk menentukan kelas dan area objek pada tingkat piksel, sedangkan keberadaan kelas pada tingkat citra tetap diprediksi oleh EfficientNet-B0.

2.9 Evaluasi Model

Evaluasi model dilakukan untuk mengukur kesesuaian antara hasil prediksi dan label acuan. Penelitian ini mengevaluasi dua model, yaitu EfficientNet-B0 untuk klasifikasi *multi-label* dan U-Net untuk segmentasi semantik multikelas. Evaluasi EfficientNet-B0 digunakan untuk membandingkan skenario tanpa segmentasi dan skenario dengan prapemrosesan segmentasi. Sementara itu, evaluasi U-Net digunakan untuk mengetahui kualitas segmentasi setiap kelas sampah pada tingkat piksel.

Pemilihan metrik disesuaikan dengan karakteristik keluaran masing-masing model. Pada klasifikasi *multi-label*, satu citra dapat memiliki lebih dari satu label sehingga evaluasi dilakukan terhadap setiap keputusan kelas secara independen. Pada segmentasi semantik multikelas, evaluasi dilakukan dengan membandingkan kelas piksel hasil prediksi dengan *ground truth mask*. Penggunaan beberapa metrik diperlukan karena setiap metrik menggambarkan aspek performa yang berbeda [31].

2.9.1 Fungsi *Loss*

Fungsi *loss* digunakan untuk mengukur tingkat kesalahan model selama proses pelatihan. Nilai *loss* yang semakin rendah menunjukkan bahwa keluaran model semakin mendekati label aktual. EfficientNet-B0 dan U-Net menggunakan fungsi *loss* yang berbeda karena keduanya menghasilkan bentuk prediksi yang berbeda. EfficientNet-B0 menghasilkan probabilitas kelas pada tingkat citra, sedangkan U-Net menghasilkan probabilitas kelas pada setiap piksel.

A *Binary Cross-Entropy*

Pada EfficientNet-B0, fungsi *loss* yang digunakan adalah *Binary Cross-Entropy* atau BCE. Fungsi ini sesuai untuk klasifikasi *multi-label* karena setiap

kelas diperlakukan sebagai keputusan biner yang independen dan diprediksi menggunakan fungsi sigmoid [32, 31]. Setiap elemen label bernilai 1 apabila kelas terdapat pada citra dan bernilai 0 apabila kelas tidak terdapat pada citra.

$$L_{BCE} = -\frac{1}{NK}\sum_{i=1}^{N}\sum_{k=1}^{K}\left[y_{i,k}\log(p_{i,k}) + (1-y_{i,k})\log(1-p_{i,k})\right] \tag{2.11}$$

Variabel N menunjukkan jumlah citra, sedangkan K menunjukkan jumlah kelas. Variabel $y_{i,k}$ merupakan label aktual citra ke-i pada kelas ke-k, sedangkan $p_{i,k}$ merupakan probabilitas yang dihasilkan model. Dalam penelitian ini, K bernilai 5. Nilai BCE meningkat apabila probabilitas prediksi semakin berbeda dari label aktual.

B *Sparse Categorical Cross-Entropy* dan *Multiclass Dice Loss*

Pada U-Net, fungsi *loss* disesuaikan dengan segmentasi semantik multikelas. Setiap piksel dipetakan ke dalam satu dari enam kelas, yaitu satu kelas latar belakang dan lima kelas sampah anorganik. Kesalahan prediksi kelas piksel dihitung menggunakan *Sparse Categorical Cross-Entropy*. Fungsi tersebut membandingkan indeks kelas aktual dengan distribusi probabilitas yang dihasilkan oleh fungsi *softmax* [33].

$$L_{SCCE} = -\frac{1}{T}\sum_{n=1}^{T}\log\left(p_{n,y_n}\right) \tag{2.12}$$

Variabel T menunjukkan jumlah piksel yang dihitung, sedangkan y_n merupakan indeks kelas aktual pada piksel ke-n. Variabel p_{n,y_n} menunjukkan probabilitas prediksi untuk kelas aktual pada piksel tersebut. Nilai *loss* meningkat apabila model memberikan probabilitas yang rendah kepada kelas yang benar. Perhitungan ini mencakup kelas objek dan kelas latar belakang.

Selain kesalahan kelas piksel, kesesuaian area segmentasi dihitung menggunakan *Multiclass Dice Loss*. Dice coefficient dihitung pada setiap kelas objek berdasarkan tumpang tindih antara probabilitas prediksi dan *ground truth mask*. Komponen Dice membantu model memperhatikan bentuk dan luas area objek, terutama ketika jumlah piksel antarkelas tidak seimbang [34, 33].

$$Dice_k = \frac{2\sum_{n=1}^{T} y_{n,k} p_{n,k} + \varepsilon}{\sum_{n=1}^{T} y_{n,k} + \sum_{n=1}^{T} p_{n,k} + \varepsilon} \tag{2.13}$$

Variabel $y_{n,k}$ menunjukkan label aktual piksel ke-n untuk kelas ke-k, sedangkan $p_{n,k}$ menunjukkan probabilitas prediksi kelas tersebut. Simbol ε merupakan nilai kecil yang digunakan untuk menghindari pembagian dengan nol. Nilai Dice yang semakin mendekati 1 menunjukkan kesesuaian area yang semakin baik. *Multiclass Dice Loss* dihitung dari rata-rata Dice lima kelas objek.

$$L_{Dice} = 1 - \frac{1}{K} \sum_{k=1}^{K} Dice_k \tag{2.14}$$

Fungsi *loss* akhir U-Net diperoleh dengan menjumlahkan *Sparse Categorical Cross-Entropy* dan *Multiclass Dice Loss*. Komponen pertama mengukur ketepatan kelas setiap piksel, sedangkan komponen kedua mengukur kesesuaian area segmentasi. Kombinasi keduanya digunakan agar model tidak hanya berfokus pada kelas piksel yang dominan. Fungsi *loss* segmentasi dirumuskan sebagai berikut.

$$L_{Seg} = L_{SCCE} + L_{Dice} \tag{2.15}$$

2.9.2 Evaluasi Model Klasifikasi Multi-Label

Evaluasi klasifikasi dilakukan untuk mengukur kemampuan EfficientNet-B0 dalam mengenali lima kelas sampah anorganik. Karena satu citra memiliki lebih dari satu label, setiap kelas diperlakukan sebagai keputusan biner yang independen. Probabilitas setiap kelas dikonversi menjadi label prediksi menggunakan nilai *threshold*. Nilai *threshold* ditentukan berdasarkan data validasi dan kemudian diterapkan tanpa perubahan pada data uji [31].

Untuk setiap kelas ke-k, hasil prediksi dikelompokkan menjadi *true positive* (TP_k), *false positive* (FP_k), *true negative* (TN_k), dan *false negative* (FN_k). Nilai tersebut menjadi dasar perhitungan *binary accuracy*, *precision*, *recall*, dan *F1-score*. Evaluasi *multi-label* juga dapat dirangkum menggunakan pendekatan *macro* dan *micro* [35, 31].

A *Binary Accuracy*

Binary accuracy mengukur proporsi keputusan label yang diprediksi dengan benar terhadap seluruh keputusan pada semua citra dan kelas. Prediksi positif dan negatif sama-sama diperhitungkan dalam metrik ini. Nilai yang semakin tinggi menunjukkan semakin banyak keputusan kelas yang sesuai dengan label aktual. Rumus *binary accuracy* ditunjukkan sebagai berikut.

$$Binary\ Accuracy = \frac{1}{NK}\sum_{i=1}^{N}\sum_{k=1}^{K}\mathbb{I}\left(y_{i,k} = \hat{y}_{i,k}\right) \tag{2.16}$$

Fungsi $\mathbb{I}$ bernilai 1 apabila label aktual sama dengan label prediksi dan bernilai 0 apabila keduanya berbeda. Variabel $\hat{y}_{i,k}$ menunjukkan label hasil prediksi setelah probabilitas dibandingkan dengan *threshold*. Metrik ini menggunakan *threshold* yang sama dengan metrik berbasis label lainnya. *Binary accuracy* digunakan sebagai informasi tambahan karena jumlah label negatif dapat lebih banyak daripada label positif.

B *Precision*, *Recall*, dan *F1-score*

Precision mengukur ketepatan model ketika memprediksi keberadaan kelas ke-k. Nilai yang tinggi menunjukkan bahwa sebagian besar prediksi positif untuk kelas tersebut sesuai dengan label aktual [31]. Rumus *precision* per kelas ditunjukkan sebagai berikut.

$$Precision_k = \frac{TP_k}{TP_k + FP_k} \tag{2.17}$$

Recall mengukur kemampuan model dalam menemukan seluruh citra yang benar-benar memuat kelas ke-k. Nilai yang tinggi menunjukkan bahwa semakin sedikit label aktual yang terlewatkan. Rumus *recall* per kelas ditunjukkan sebagai berikut.

$$Recall_k = \frac{TP_k}{TP_k + FN_k} \tag{2.18}$$

F1-score merupakan rata-rata harmonis antara *precision* dan *recall*. Metrik ini digunakan untuk mengukur keseimbangan antara ketepatan prediksi positif

dan kemampuan model menemukan label positif. Rumus *F1-score* per kelas ditunjukkan sebagai berikut.

$$F1_k = 2\frac{Precision_k \times Recall_k}{Precision_k + Recall_k} \tag{2.19}$$

Nilai *precision*, *recall*, dan *F1-score* dihitung secara terpisah untuk setiap kelas. Hasil tersebut digunakan untuk mengetahui kelas yang masih sulit dikenali oleh model. Evaluasi per kelas juga menunjukkan apakah segmentasi memberikan pengaruh yang berbeda pada setiap jenis sampah. Nilai per kelas kemudian dirangkum menggunakan pendekatan *macro* dan *micro*.

C *Macro F1-score*

Macro F1-score dihitung dengan merata-ratakan nilai *F1-score* seluruh kelas. Setiap kelas memperoleh bobot yang sama tanpa dipengaruhi oleh jumlah kemunculannya. Metrik ini digunakan untuk mengetahui keseimbangan performa model pada lima kelas sampah [35, 31]. Rumusnya ditunjukkan sebagai berikut.

$$Macro\ F1 = \frac{1}{K}\sum_{k=1}^{K} F1_k \tag{2.20}$$

D *Micro Precision*, *Micro Recall*, dan *Micro F1-score*

Pendekatan *micro average* menggabungkan seluruh nilai TP, FP, dan FN dari semua kelas sebelum metrik dihitung. Pendekatan ini menggambarkan performa keseluruhan terhadap seluruh keputusan label. Kelas yang lebih sering muncul dapat memberikan pengaruh lebih besar terhadap nilai *micro average*. Rumus *micro precision* ditunjukkan sebagai berikut.

$$Micro\ Precision = \frac{\sum_{k=1}^{K} TP_k}{\sum_{k=1}^{K}(TP_k + FP_k)} \tag{2.21}$$

Micro recall mengukur proporsi seluruh label positif yang berhasil dikenali. Nilainya dihitung menggunakan gabungan *true positive* dan *false negative* dari seluruh kelas.

$$Micro\ Recall = \frac{\sum_{k=1}^{K} TP_k}{\sum_{k=1}^{K} (TP_k + FN_k)} \tag{2.22}$$

Micro F1-score dihitung dari *micro precision* dan *micro recall*. Metrik ini menunjukkan keseimbangan performa model terhadap seluruh keputusan label pada dataset.

$$Micro\ F1 = 2\frac{Micro\ Precision \times Micro\ Recall}{Micro\ Precision + Micro\ Recall} \tag{2.23}$$

E *Macro Area Under the Curve*

Area Under the Curve atau AUC mengukur kemampuan model dalam membedakan label positif dan negatif pada berbagai nilai *threshold*. AUC dihitung berdasarkan kurva *Receiver Operating Characteristic* untuk setiap kelas. Nilai AUC yang semakin mendekati 1 menunjukkan kemampuan pemisahan kelas yang semakin baik. *Macro AUC* diperoleh dengan merata-ratakan nilai seluruh kelas [31, 35].

$$Macro\ AUC = \frac{1}{K}\sum_{k=1}^{K} AUC_k \tag{2.24}$$

Variabel AUC_k menunjukkan nilai AUC kelas ke-k. Berbeda dengan metrik berbasis label, AUC tidak hanya menilai hasil pada satu nilai *threshold*. Metrik ini digunakan sebagai informasi tambahan untuk melihat kemampuan pemisahan probabilitas model. Nilainya tetap dianalisis bersama metrik berbasis label prediksi.

F *Mean Average Precision*

Mean Average Precision atau mAP digunakan untuk menilai kualitas pemeringkatan probabilitas pada klasifikasi *multi-label*. *Average Precision* merangkum hubungan antara *precision* dan *recall* pada berbagai nilai *threshold*. Nilai AP dihitung pada setiap kelas dan kemudian dirata-ratakan [36]. Rumus mAP ditunjukkan sebagai berikut.

$$mAP = \frac{1}{K}\sum_{k=1}^{K} AP_k \tag{2.25}$$

Variabel AP_k menunjukkan *Average Precision* kelas ke-k. Nilai mAP yang semakin tinggi menunjukkan bahwa model semakin baik dalam memberikan probabilitas tinggi kepada label yang benar. Metrik ini tidak bergantung pada satu nilai *threshold*. Dalam penelitian ini, mAP digunakan bersama *macro F1-score* dan *micro F1-score*.

G *Hamming Loss*

Hamming loss mengukur proporsi kesalahan label terhadap seluruh keputusan label. Kesalahan dapat berupa kelas aktual yang tidak diprediksi atau kelas yang tidak terdapat pada citra tetapi diprediksi aktif. Nilai *Hamming loss* yang semakin rendah menunjukkan semakin sedikit kesalahan label [31, 35]. Rumusnya ditunjukkan sebagai berikut.

$$Hamming\ Loss = \frac{1}{NK}\sum_{i=1}^{N}\sum_{k=1}^{K} \mathbb{I}\left(y_{i,k} \neq \hat{y}_{i,k}\right) \tag{2.26}$$

H *Subset Accuracy*

Subset accuracy menganggap prediksi sebuah citra benar apabila seluruh label prediksi sama dengan seluruh label aktual. Satu kesalahan label menyebabkan prediksi citra tersebut dianggap tidak sepenuhnya benar. Metrik ini lebih ketat dibandingkan metrik yang dihitung pada setiap kelas [35]. Rumus *subset accuracy* ditunjukkan sebagai berikut.

$$Subset\ Accuracy = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}(\mathbf{y}_i = \hat{\mathbf{y}}_i) \tag{2.27}$$

I *Confusion Matrix* Biner per Kelas

Analisis kesalahan dilakukan menggunakan *confusion matrix* biner untuk setiap kelas. Setiap matriks memuat TP_k, FP_k, TN_k, dan FN_k pada satu kelas.

Pendekatan ini digunakan karena sebuah citra dapat memiliki lebih dari satu label sehingga satu *confusion matrix* multikelas tidak dapat mewakili seluruh prediksi. Matriks per kelas menunjukkan label positif yang salah diprediksi maupun label aktual yang terlewatkan.

Metrik utama yang digunakan untuk membandingkan kedua skenario adalah *macro F1-score*, *micro F1-score*, mAP, *Hamming loss*, dan *subset accuracy*. Nilai *precision*, *recall*, serta *F1-score* per kelas digunakan untuk memperinci performa setiap kelas. *Binary accuracy* dan *macro AUC* digunakan sebagai informasi tambahan. Seluruh metrik berbasis label dihitung menggunakan *threshold* yang ditentukan pada data validasi.

2.9.3 Evaluasi Model Segmentasi

Evaluasi segmentasi dilakukan untuk mengukur kesesuaian antara *mask* semantik multikelas hasil prediksi U-Net dan *ground truth mask*. Evaluasi dilakukan secara terpisah pada setiap kelas karena keluaran model terdiri dari satu kelas latar belakang dan lima kelas objek. Metrik yang digunakan meliputi *pixel precision*, *pixel recall*, IoU, dan Dice coefficient. Kinerja keseluruhan lima kelas objek dirangkum menggunakan mIoU dan *macro Dice* [37, 33].

Untuk setiap kelas ke-k, prediksi piksel dikelompokkan menjadi TP_k^{pixel}, FP_k^{pixel}, TN_k^{pixel}, dan FN_k^{pixel}. TP_k^{pixel} menunjukkan jumlah piksel kelas ke-k yang diprediksi dengan benar. FP_k^{pixel} menunjukkan piksel dari kelas lain yang salah diprediksi sebagai kelas ke-k. Sementara itu, FN_k^{pixel} menunjukkan piksel kelas ke-k yang tidak berhasil dikenali.

A *Pixel Precision* dan *Pixel Recall*

Pixel precision mengukur ketepatan prediksi piksel pada kelas ke-k. Nilai yang tinggi menunjukkan bahwa sebagian besar piksel yang diprediksi sebagai kelas tersebut sesuai dengan kelas aktual. Rumusnya ditunjukkan sebagai berikut.

$$Pixel\ Precision_k = \frac{TP_k^{pixel}}{TP_k^{pixel} + FP_k^{pixel}} \tag{2.28}$$

Pixel recall mengukur kemampuan model dalam menemukan seluruh piksel yang sebenarnya termasuk kelas ke-k. Nilai yang rendah menunjukkan

bahwa sebagian area aktual kelas tersebut tidak berhasil dipertahankan pada hasil segmentasi. Rumusnya ditunjukkan sebagai berikut.

$$Pixel\ Recall_k = \frac{TP_k^{pixel}}{TP_k^{pixel} + FN_k^{pixel}} \tag{2.29}$$

B *Intersection over Union*

Intersection over Union atau IoU mengukur perbandingan antara area irisan dan area gabungan hasil prediksi dengan *ground truth*. Metrik ini dihitung secara terpisah untuk setiap kelas. Nilai yang semakin mendekati 1 menunjukkan kesesuaian area yang semakin baik [37]. Rumus IoU untuk kelas ke-k ditunjukkan sebagai berikut.

$$IoU_k = \frac{|M_{p,k} \cap M_{g,k}|}{|M_{p,k} \cup M_{g,k}|} \tag{2.30}$$

Variabel $M_{p,k}$ menunjukkan area yang diprediksi sebagai kelas ke-k, sedangkan $M_{g,k}$ menunjukkan area aktual kelas tersebut. Simbol $\cap$ menunjukkan area irisan, sedangkan $\cup$ menunjukkan area gabungan. Hasil per kelas digunakan untuk mengetahui perbedaan kemampuan U-Net dalam melakukan segmentasi setiap jenis sampah.

C Dice Coefficient

Dice coefficient mengukur tingkat kesamaan antara area prediksi dan area aktual dengan memberikan bobot dua kali pada area irisan. Nilainya berada pada rentang 0 sampai 1. Nilai yang mendekati 1 menunjukkan hasil segmentasi yang semakin sesuai dengan *ground truth mask* [37]. Rumus Dice per kelas ditunjukkan sebagai berikut.

$$Dice_k = \frac{2|M_{p,k} \cap M_{g,k}|}{|M_{p,k}| + |M_{g,k}|} \tag{2.31}$$

D *Mean Intersection over Union* dan *Macro Dice*

Nilai IoU dan Dice dihitung pada lima kelas objek, kemudian dirata-ratakan untuk memperoleh mIoU dan *macro Dice*. Setiap kelas memperoleh bobot yang sama sehingga nilai rata-rata tidak hanya dipengaruhi oleh kelas dengan area piksel yang lebih besar. Kelas latar belakang dilaporkan secara terpisah agar hasil utama lebih menggambarkan kemampuan segmentasi objek [31, 33].

$$mIoU = \frac{1}{K}\sum_{k=1}^{K} IoU_k \tag{2.32}$$

$$Macro\ Dice = \frac{1}{K}\sum_{k=1}^{K} Dice_k \tag{2.33}$$

Variabel K menunjukkan lima kelas objek sampah anorganik. Hasil IoU, Dice, *pixel precision*, dan *pixel recall* dilaporkan untuk setiap kelas. Nilai mIoU dan *macro Dice* digunakan untuk merangkum kualitas segmentasi secara keseluruhan. Hasil tersebut kemudian digunakan untuk menganalisis hubungan antara kualitas *mask* dan kinerja klasifikasi pada skenario dengan segmentasi.