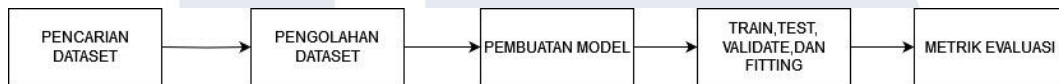


BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1 Perancangan Sistem



Gambar 3.1 Flowchart perancangan sistem

Penelitian ini akan dibagi menjadi 4 tahapan, yaitu tahap pencarian dataset, tahap pembuatan model, tahap train, test, validate, dan tahap dokumentasi serta hasil. Gambar titik 3.1 merupakan penjabaran secara garis besar mengenai metode yang dipakai.

Penelitian akan dilakukan pada environment google colab berbasis *cloud computing* dengan menggunakan bahasa pemrograman Python untuk melakukan pembuatan model, model juga akan dilengkapi oleh library pendukung lainnya.

3.2 Pencarian Dataset

Dalam tahap pencarian Dataset, penulis menggunakan media internet sebagai dasar dalam menemukan gambar. Gambar tersebut berdasar pada kriteria gambar sampah botol plastik sebagai wakil dari kategori plastik serta gambar kardus sebagai wakil dari kategori plastik. Gambar gambar tersebut didapatkan dari berbagai macam sumber yang telah dicantumkan pada daftar pustaka [32][33][34] yang di unduh secara *Random* atau acak. Hasil dari pencarian gambar-gambar tersebut dikumpulkan hingga berisi 4000 gambar.

Terdapat beberapa *dataset overlap* yang ditemukan. *Dataset overlap* merupakan situasi dimana output gambar, mempunyai kesamaan atau tumpang tindih dalam menentukan validasi pengujian. Situasi ini dapat mengakibatkan peningkatan kinerja dan stabilitas model hingga menghasilkan transfer

pembelajaran dengan baik serta membuat efisiensi pelatihan pada model. Namun, hal tersebut dapat membuat model menjadi *overfitting*. Penulis memutuskan untuk hanya mentrain layer terakhir untuk menghindari hal ini

3.3 Pengolahan Dataset

Semua gambar tersebut kemudian diinput kedalam roboflow untuk diukur rasionya. Gambar tersebut kemudian dipecah menjadi 2, kelas plastik dan non plastik. Disini penulis membagi sama rata masing masing 2000 per kelasnya. Rasio untuk membuat batch training pun akhirnya ditentukan menjadi 80/10/10, sehingga jika dihitung lebih lanjut maka terdapat 1600 perkelas gambarnya untuk training, 200 gambar perkelasnya untuk test, dan 200 gambar per kelas untuk validasi. Hitungan terperinci terlampir di bawah ini.

Tabel 3.1 Total gambar pada dataset

	Training	Test	Validation	Total
Plastik	1600	200	200	2000
Non Plastik	1600	200	200	2000
Total	3200	400	400	4000

Umumnya tidak ada aturan dan alasan yang baku mengenai splitting pada rasio diatas serta jumlah dataset yang diperlukan. Namun, dataset yang lebih besar dapat membantu membuat training set yang lebih besar sehingga dapat membantu model untuk memahami hubungan kompleks antara fitur dan variable dari target yang ditentukan. Biasanya, praktik yang umum dilakukan adalah membagi dataset menjadi 60-80% training, 10-20% test, dan 10-20% validation[35].

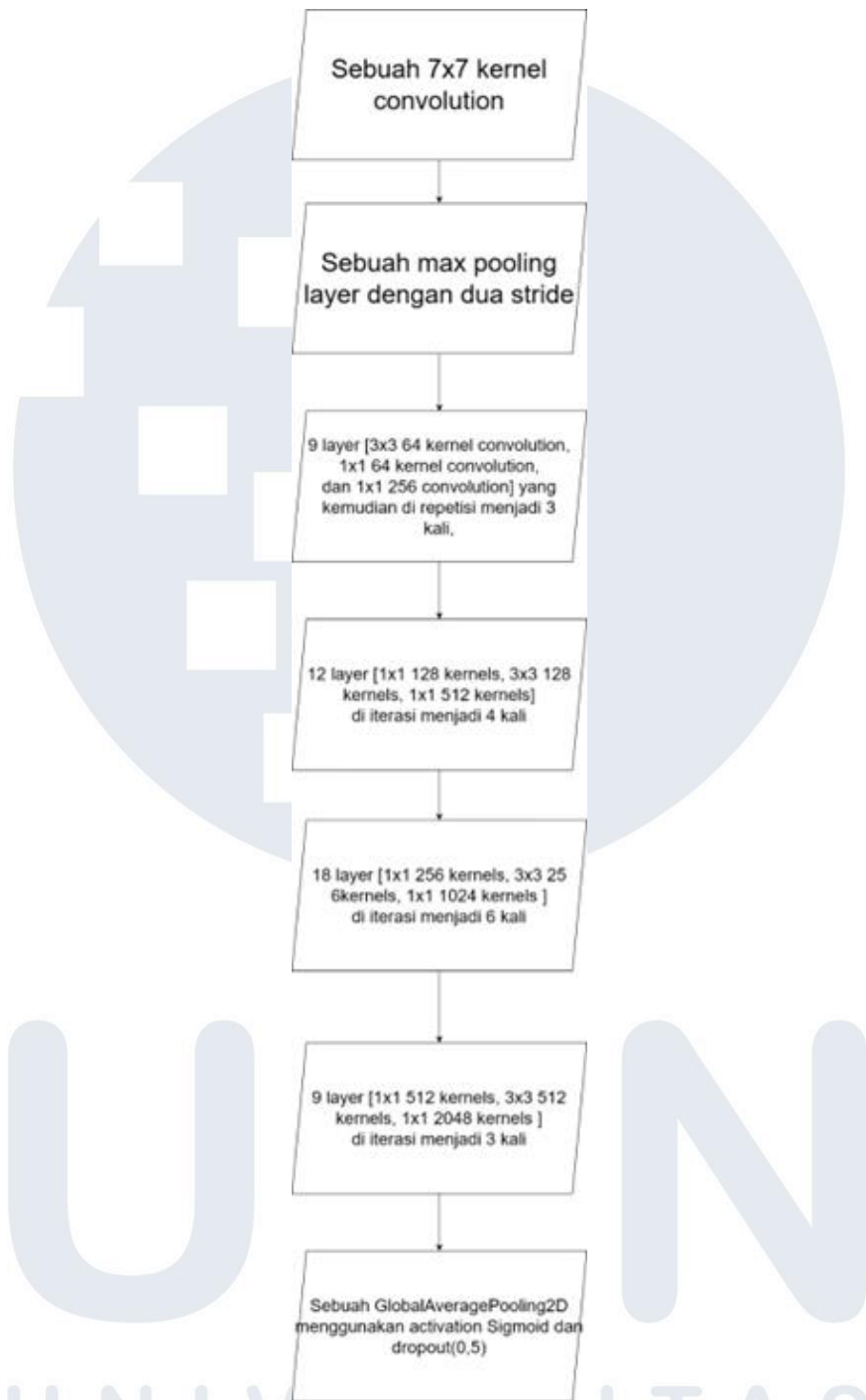
Setelah dipecah, gambar tersebut di resize menjadi resolusi 224x224 guna menyesuaikan input pada ResNet50.

3.5 Pembuatan Model

Penelitian dilakukan dengan ResNet-50. Selaras dengan bab 2, model ResNet-50 akan terbentuk seperti ini:

- Sebuah 7x7 kernel convolution digunakan untuk mengekstraksi fitur-fitur awal gambar input.
- Sebuah max pooling layer dengan dua stride digunakan untuk mengurangi dimensi spasial dari fitur yang sudah dibuat pada *layer convolution*.
- 9 layer [3x3 64 kernel convolution, 1x1 64 kernel convolution, dan 1x1 256 convolution] yang kemudian di repetisi menjadi 3 kali. Digunakan untuk menyeleksi lebih dalam terkait dengan *feature* gambar.
- 12 layer [1x1 128 kernels, 3x3 128 kernels, 1x1 512 kernels] di iterasi menjadi 4 kali. Digunakan untuk menyeleksi lebih dalam terkait dengan *feature* gambar.
- 18 layer [1x1 256 kernels, 3x3 256 kernels, 1x1 1024 kernels] di iterasi menjadi 6 kali. Digunakan untuk menyeleksi lebih dalam terkait dengan *feature* gambar.
- 9 layer [1x1 512 kernels, 3x3 512 kernels, 1x1 2048 kernels] di iterasi menjadi 3 kali. Digunakan untuk menyeleksi lebih dalam terkait dengan *feature* gambar. Terdapat 48 convolutional residual block terhitung dari atas yang membedakan arsitektur ResNet50 dengan arsitektur lainnya.
- Sebuah average pooling menggunakan activation softmax. Untuk menyeleksi input berdasarkan *class*.

Seperti pada gambar 3.2, penulis membekukan seluruh layer di atasnya dan menyisakan layer akhir di bawah model dalam penelitian ini. Pada Layer akhir ditambahkan pula parameter dropout() dan averagepooling yang diganti menjadi GlobalAveragePooling2D. Semua hal ini disetting karena untuk menghindari training yang lama serta menghindari overfitting yang ada[37][38][39]. Tak lupa dense layer juga diset sigmoid serta dibuat 2 kelas pada akhir layer karena penelitian hanya membutuhkan 2 kelas. Disimpulkan bahwa arsitektur akan menjadi seperti ini:



Gambar 3.2 Model Arsitektur versi penulis

3.6 Training, Testing dan Validation

Tahap ini berisi tentang cara bagaimana model dan dataset digabung. Model akan difitting dengan optimizer adam yang merupakan default serta binary categorical untuk memungkinkan 2 kelas label di training. Terdapat penggunaan *hyperparameter* pada model. Hyperparameter pada ResNet-50 memiliki peran penting dalam mengoptimalkan performa model dan meminimalkan overfitting[5][19][36]. Konfigurasi Hyperparameter ini berdasar *default* pada website keras.io[40] .

- Epochs digunakan untuk menentukan berapa kali seluruh dataset akan diproses oleh model. Epochs dapat mempengaruhi konvergensi model dan performa model.
- Steps Per Epoch digunakan untuk menentukan jumlah batch yang akan diproses pada setiap epoch. Steps_per_epoch dapat mempengaruhi kecepatan training dan performa model.
- Batch_size digunakan untuk menentukan jumlah sampel data yang diproses pada setiap iterasi. Batch_size dapat mempengaruhi kecepatan training dan performa model.
- Verbose digunakan untuk menampilkan output dari proses training. Verbose dapat membantu dalam memantau proses training dan mengevaluasi performa model.
- Validation_data digunakan untuk mengevaluasi performa model pada data validasi. Validation_data dapat membantu dalam mengevaluasi performa model dan meminimalkan overfitting.
- Callbacks digunakan untuk menambahkan fungsi tambahan pada proses training, seperti early stopping. Callbacks dapat membantu dalam mengoptimalkan performa model dan meminimalkan overfitting.

Sementara dalam penggunaan optimizer, terdapat konfigurasi hyperparameter sebagai berikut[40]. untuk mengoptimalkan performa model dan meminimalkan overfitting pada ResNet-50.

- Learning rate adalah hyperparameter yang menentukan seberapa besar langkah yang diambil oleh model saat memperbarui bobotnya. Learning rate yang terlalu besar dapat menyebabkan model tidak konvergen, sedangkan learning rate yang terlalu kecil dapat menyebabkan model konvergen terlalu lambat.
- Epsilon adalah hyperparameter yang digunakan untuk mencegah pembagian dengan nol pada optimizer Adam. Epsilon yang terlalu kecil dapat menyebabkan pembagian dengan nol, sedangkan epsilon yang terlalu besar dapat menyebabkan model tidak konvergen.
- Beta_1 dan beta_2 adalah hyperparameter yang menentukan momentum pada optimizer Adam. Beta_1 dan beta_2 yang terlalu besar dapat menyebabkan model tidak konvergen, sedangkan beta_1 dan beta_2 yang terlalu kecil dapat menyebabkan model konvergen terlalu lambat.
- Amsgrad adalah hyperparameter yang digunakan untuk mengontrol penggunaan momentum pada optimizer Adam. Amsgrad yang diaktifkan dapat membantu model konvergen lebih cepat dan menghasilkan hasil yang lebih baik.

3.7 Metrik Evaluasi

Ini merupakan metode terakhir dimana hasil akan diukur dengan metrik evaluasi yang ditentukan dengan output berupa nilai Accuracy, Precision, Recall, F1-Score. Hasil yang didapatkan juga merupakan trial and error bagi penelitian.

U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A