



PENERAPAN *DEEP LEARNING* PADA KLASIFIKASI CITRA SAMPAH DENGAN METODE *CONVOLUTIONAL NEURAL NETWORK*

Disusun Oleh :

Rheza Pandya Andhikaputra - 140810200023

Pembimbing Utama

Dr. Intan Nurma Yulita, M.T
NIP. 198507042015042003

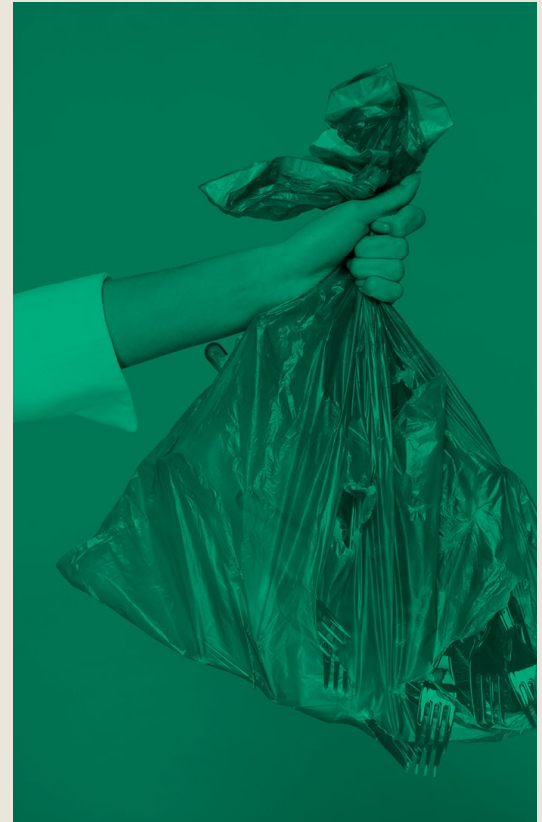
Co-Pembimbing

Drs. Akik Hidayat, M.Kom
NIP. 196110181986031002

LATAR BELAKANG

Jumlah limbah padat kota yang dihasilkan secara global mencapai **2,01** miliar ton per tahun, dan sekitar **33%** dari jumlah tersebut tidak dikelola dengan cara yang ramah lingkungan (World Bank, 2018)

Jumlah timbulan sampah di Indonesia mencapai **33,27** juta ton per tahun, namun hanya **63,53%** atau sekitar **21,1** juta ton sampah yang berhasil dikelola dengan **38,22%** diantaranya merupakan sampah rumah tangga (Direktorat Penanganan Sampah, 2022)



IDENTIFIKASI MASALAH

1

Bagaimana cara membuat model *Convolutional Neural Network* untuk klasifikasi citra sampah?

2

Bagaimana tahap konfigurasi *hyperparameter* pada model *Convolutional Neural Network* yang terbaik sehingga diperoleh hasil *F1 Score* yang optimal?

3

Bagaimana cara menerapkan model *Convolutional Neural Network* pada aplikasi berbasis *website*?

BATASAN MASALAH

1

Metode yang digunakan adalah *Convolutional Neural Network* menggunakan *framework* TensorFlow dengan bahasa pemrograman Python

2

Hyperparameter yang dibandingkan dalam skenario pelatihan yaitu penggunaan *Pretrained Model* dengan arsitektur ResNet, VGG, atau DenseNet, penggunaan *Optimizer*, dan penggunaan *Dropout Layer*

3

Data citra yang digunakan merupakan *dataset* Garbage Classification yang diperoleh dari *platform* Kaggle dan telah memiliki label sebanyak 6 kelas, yaitu *Cardboard*, *Glass*, *Metal*, *Paper*, *Plastic*, dan *Trash*

4

Tahap pembuatan model dan pelatihan data dilakukan dengan menggunakan *platform* Google Colab dengan mengimpor *library* yang dibutuhkan.

MAKSUD PENELITIAN

Membuat suatu sistem klasifikasi citra sampah dengan menggunakan metode *Convolutional Neural Network* dengan nilai *F1 Score* yang optimal

TUJUAN PENELITIAN

1. Dapat mengimplementasikan metode *Convolutional Neural Network* untuk klasifikasi citra sampah.
2. Dapat menemukan konfigurasi *hyperparameter* pada model *Convolutional Neural Network* dengan mendapatkan nilai *F1 Score* yang paling optimal untuk mendeteksi citra sampah.
3. Dapat menghasilkan sebuah aplikasi klasifikasi citra sampah berbasis *website*.

MANFAAT PENELITIAN

1

Dapat memudahkan pengguna yang ingin mengklasifikasikan sampah

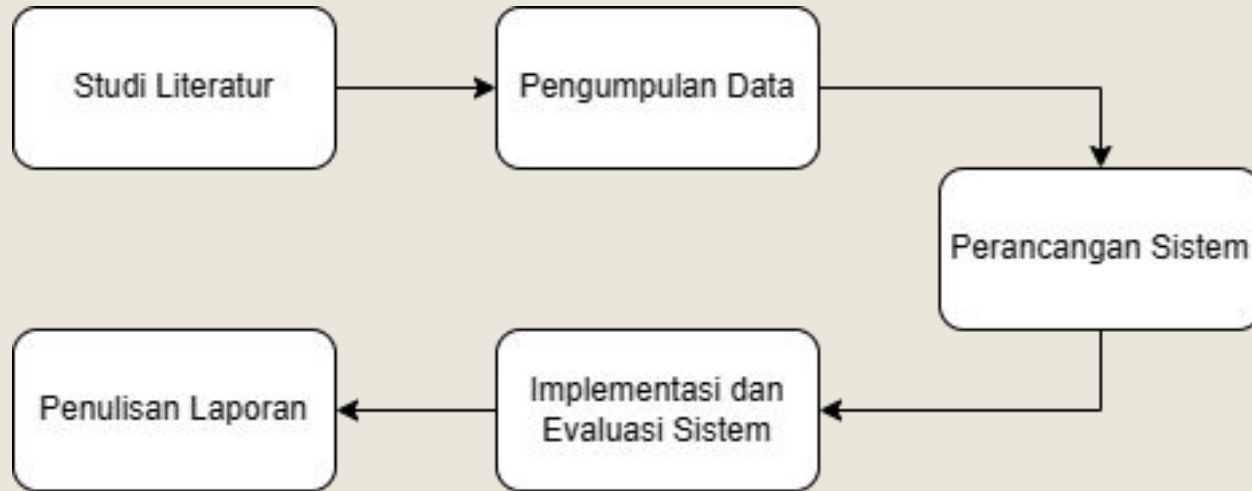
2

Dapat menambah pengetahuan mengenai implementasi metode *Convolutional Neural Network* pada klasifikasi citra sampah

3

Dapat mengembangkan penelitian yang sebelumnya telah dibuat

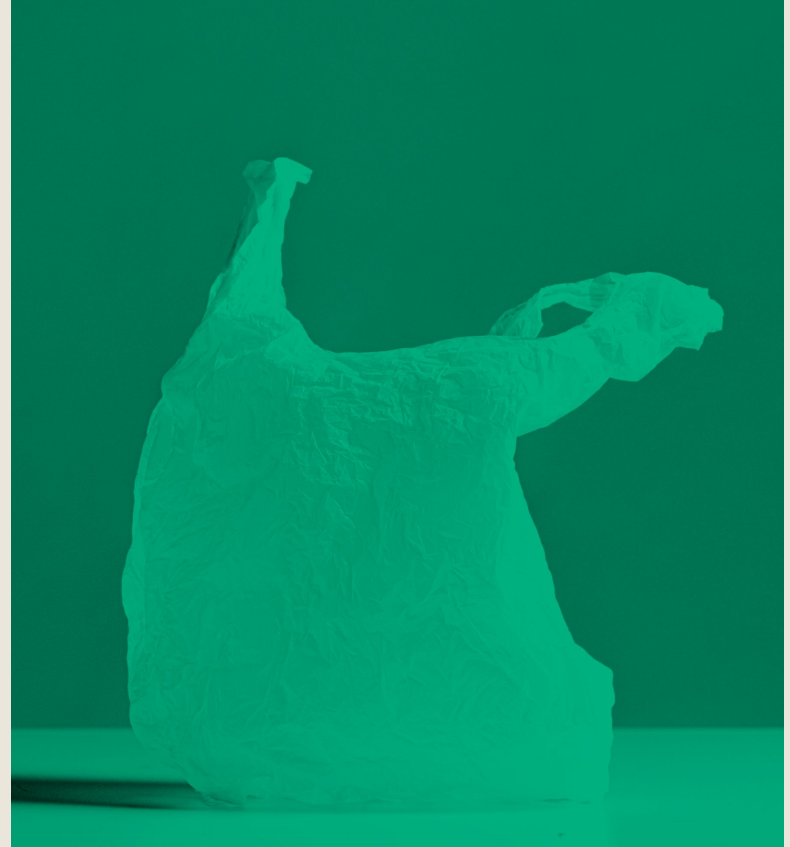
METODOLOGI PENELITIAN



APA ITU SAMPAH?

Sampah adalah materi yang tersisa dari aktivitas sehari-hari manusia atau proses alam yang berbentuk padat (Undang-Undang No.2 Tahun 2008)

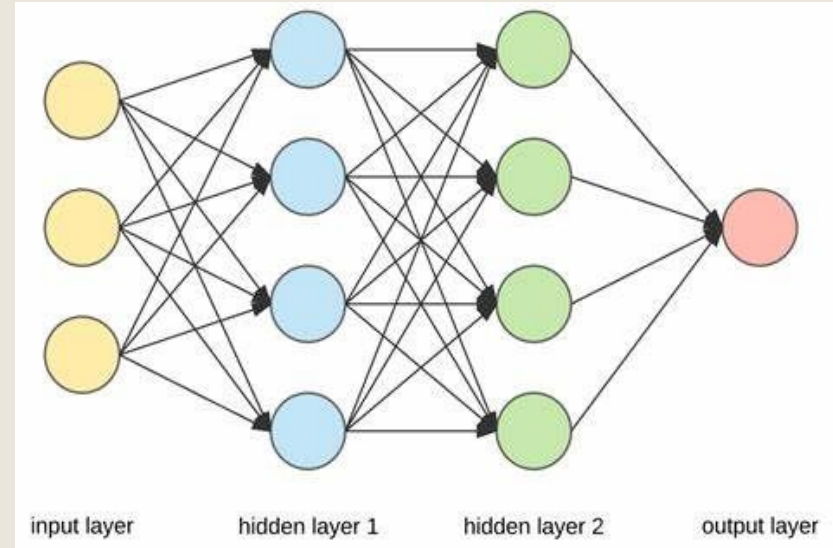
Sampah dapat diklasifikasikan berdasarkan bahan dasarnya, seperti kardus, plastik, besi, kaca, dan lain-lain



APA ITU CONVOLUTIONAL NEURAL NETWORK ?

Convolutional Neural Network (CNN) merupakan sebuah arsitektur model yang sangat populer dan efektif dalam dalam pengaplikasian pada bidang *Deep Learning*

Model ini telah banyak diaplikasikan dalam bidang *image classification*



HYPERPARAMETER CONVOLUTIONAL NEURAL NETWORK

Convolutional Neural Network memiliki banyak *hyperparameter* yang diterapkan pada model, berikut *hyperparameter* yang dibandingkan pada penelitian ini :

1

Pretrained Model

ResNet50
VGG19
DenseNet121

2

Optimizers

Adam
Nadam
AdamW

3

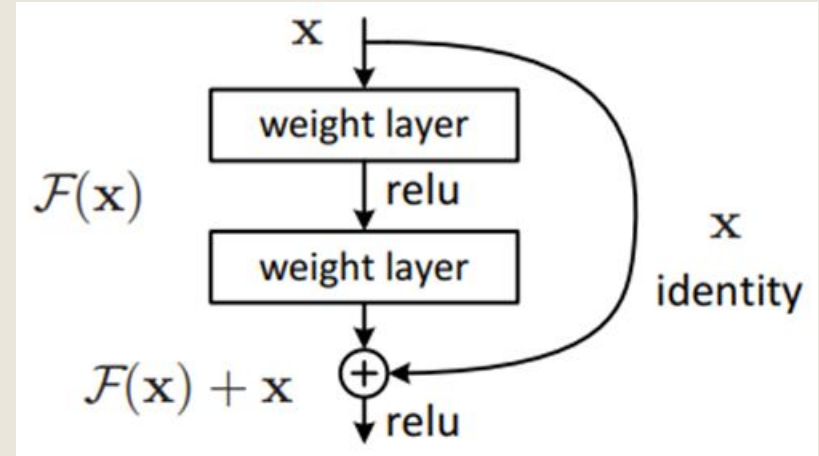
Dropout Layer

No Dropout
20% Dropout
50% Dropout

Residual Network (ResNet)

Residual Network (ResNet) dibuat untuk mengatasi permasalahan degradasi akurasi pada tingkat kedalaman pada *Convolutional Neural Network* (He et al., 2016)

Pada ResNet, terdapat *shortcut connection* untuk memudahkan *identity mapping* dengan menambahkan *output* dari *layer* sebelumnya ke *output layer* selanjutnya



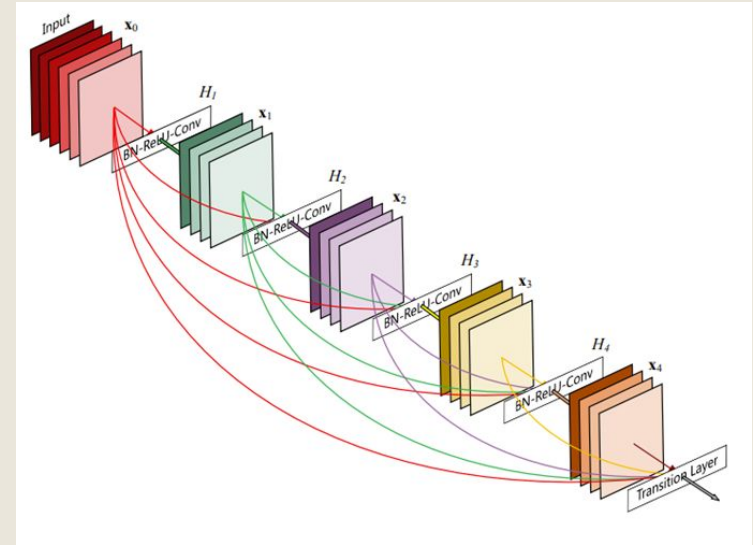
Visual Geometry Group (VGG)

Visual Geometry Group (VGG) dibuat dengan arsitektur *Convolutional Neural Network* tradisional dengan memanfaatkan *layer convolution* dan *pooling* (Simonyan & Zisserman, 2015)

Densely Connected Convolutional Networks (DenseNet)

Densely Connected Convolutional Networks (DenseNet) merupakan pengembangan dari ResNet

DenseNet memiliki koneksi yang menghubungkan setiap *layer* dengan *layer-layer* berikutnya dengan menggabungkan *feature map* (Huang et al., 2017)



Adaptive Moment Estimation (Adam)

Adam merupakan *Optimization Algorithm* yang memanfaatkan perkiraan adaptif dari momen orde rendah untuk tujuan stokastik (Kingma & Ba, 2014)

Metode ini menghitung *learning rate* secara adaptif untuk setiap parameter berdasarkan perkiraan momen pertama dan kedua dari gradien

Nesterov-accelerated Adaptive Moment Estimation

Nesterov-accelerated Adaptive Moment Estimation (Nadam) merupakan modifikasi dari Adam dengan mengganti komponen momentum menggunakan algoritma *Nesterov's Accelerated Gradient* (NAG) (Dozat, 2016)

Dengan penggantian komponen momentum dengan mengadopsi NAG dapat meningkatkan kecepatan konvergensi dengan mencegah *overshoot*

Adaptive Moment Estimation with Weight Decay

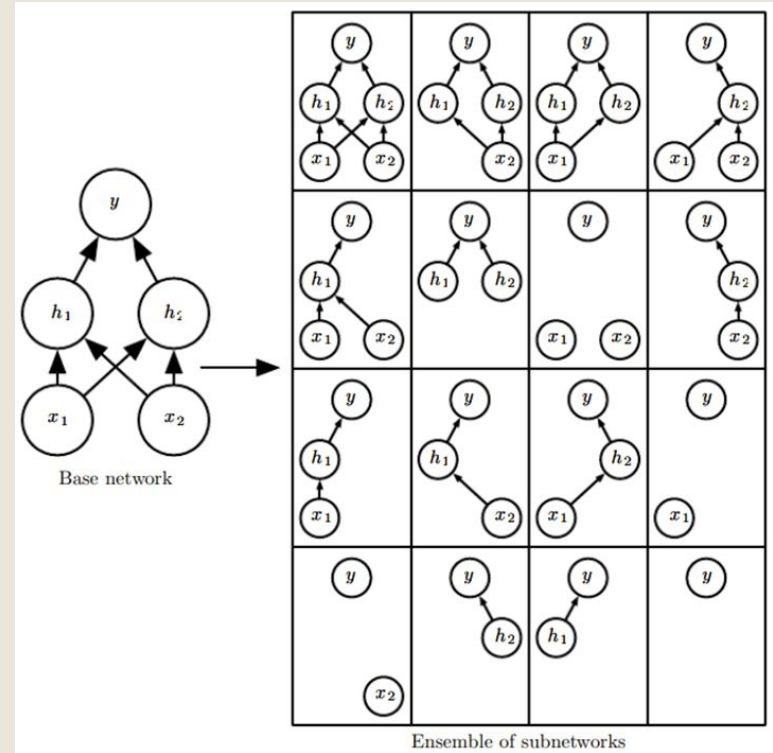
Adaptive Moment Estimation with Weight Decay (AdamW) merupakan modifikasi dari Adam dengan mengintegrasikan Weight Decay untuk meningkatkan efektivitas regularisasi pada pelatihan model (Loshchilov & Hutter, 2019)

AdamW melibatkan *Weight Decay* pada pembaruan gradien dengan tujuan mencegah *overfitting* sehingga dapat meningkatkan generalisasi pada data-data baru terutama dalam *image recognition*

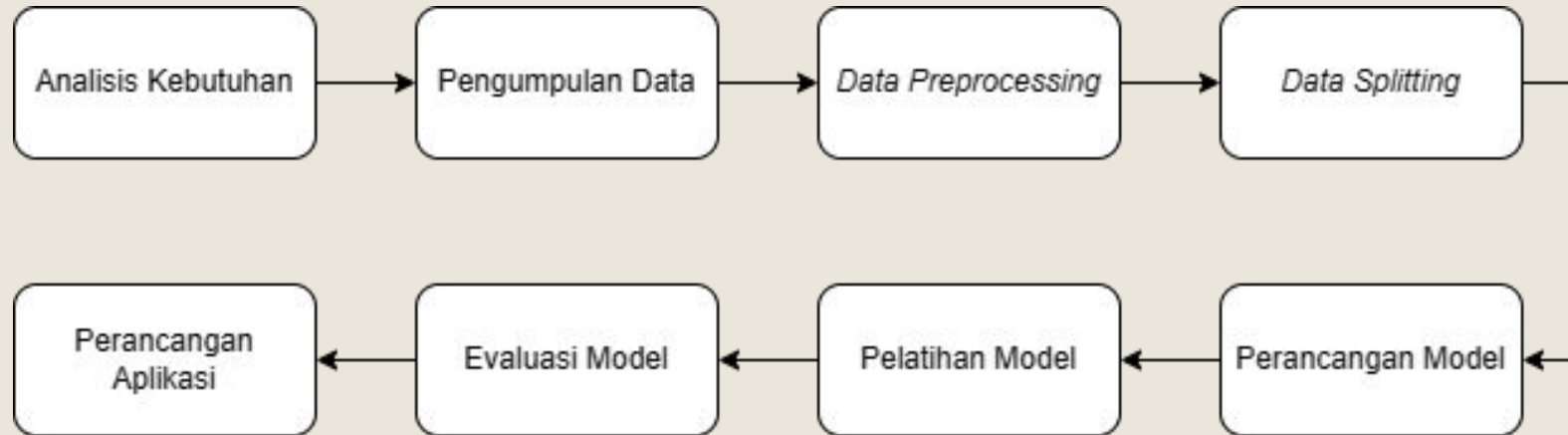
Dropout

Dropout merupakan teknik untuk mengurangi kesalahan generalisasi pada *training* dengan melatih *ensemble* yang terdiri dari semua sub-jaringan yang dapat dibangun dengan menghapus *unit non-output* dari jaringan dasar (Goodfellow et al., 2016)

Dropout digunakan dengan memberikan variabel probabilitas untuk mengaktifkan dan menonaktifkan *node* dalam suatu *layer*



ANALISIS DAN PERANCANGAN



KEBUTUHAN PERANGKAT KERAS

Laptop dengan spesifikasi :

- Manufaktur : Lenovo Legion 5
- Prosesor : Intel Core i7 Gen 10
- RAM : 16 GB
- VGA : NVIDIA GeForce GTX 1660 Ti

KEBUTUHAN PERANGKAT LUNAK

Software :

1. Python 3.8
2. Kaggle
3. Figma
4. Google Colab, dengan spesifikasi :
 - Nama : Python 3 Google Compute Engine Backend
 - RAM : 12.7 GB
 - GPU Device : Tesla T4
 - GPU Name : NVIDIA-SMI
 - GPU RAM : 15 GB
 - CUDA : 12.0

KEBUTUHAN PERANGKAT LUNAK

Framework yang digunakan :

- TensorFlow
- Laravel

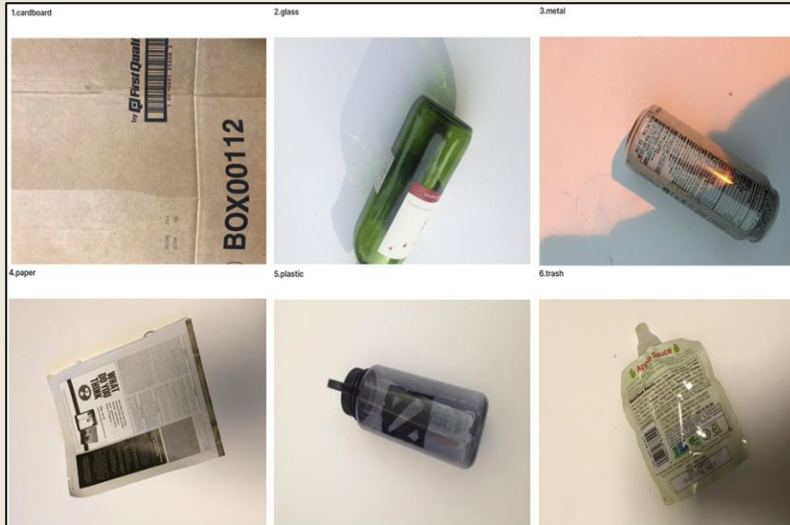
Library yang digunakan :

- OpenCV
- Numpy
- Matplotlib
- Scikit-learn
- TensorFlow Keras
- Flask

PENGUMPULAN DATA

Dataset diambil dari *platform* Kaggle yang bernama Garbage Classification.

Link *dataset* : <https://www.kaggle.com/datasets/asdasdasdasdas/garbage-classification>



Komposisi *dataset*:

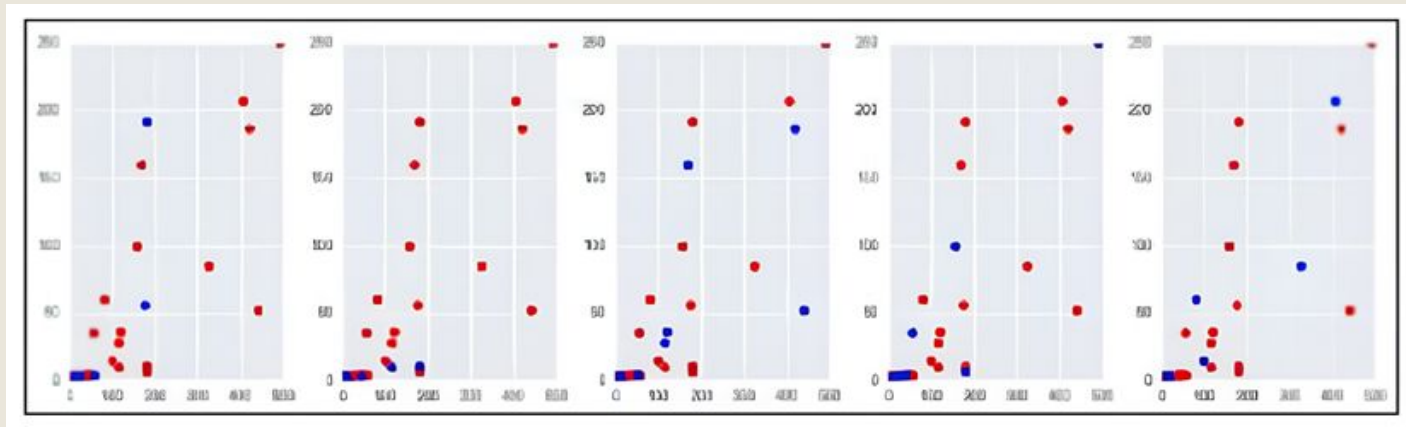
- *Cardboard* : 403 citra
- *Glass* : 501 citra
- *Metal* : 410 citra
- *Plastic* : 594 citra
- *Paper* : 482 citra
- *Trash* : 137 citra
- Jumlah data : 2527 citra

DATA PREPROCESSING

Tahap	Detail	Parameter
<i>Image Resize</i>	<i>Image Resize</i>	224x224
<i>Pretrained Model Preprocess Input</i>	<i>Pretrained Model Preprocess Input</i> sesuai dengan <i>Pretrained Model</i> yang digunakan	Resnet50.preprocess_input, vgg19.preprocess_input, atau densenet.preprocess_input
<i>Image Augmentation</i>	<i>Horizontal Flip</i>	TRUE
	<i>Vertical Flip</i>	TRUE
	<i>Rotation Range</i>	20
	<i>Shear Range</i>	0.2
	<i>Zoom Range</i>	0.2
	<i>Width Shift Range</i>	0.1
	<i>Height Shift Range</i>	0.1

DATA SPLITTING

Data Splitting dilakukan dengan metode *5-Fold Cross Validation* dengan ilustrasi berikut :



● *Data Training*

● *Data Testing*

SKENARIO PELATIHAN MODEL

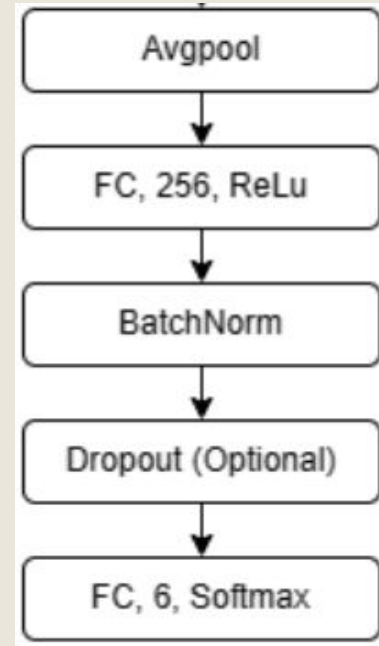
Skenario perbandingan *hyperparameter* dilakukan dengan kombinasi 3 *hyperparameter* dengan masing-masing 3 skenario dengan total skenario sebanyak 27 skenario

Skenario	Pretrained Model	Optimizers	Dropout Layer
1	ResNet50	Adam	No Dropout
2	ResNet50	Adam	0.2
3	ResNet50	Adam	0.5
4	ResNet50	Nadam	No Dropout
5	ResNet50	Nadam	0.2
6	ResNet50	Nadam	0.5
7	ResNet50	AdamW	No Dropout
8	ResNet50	AdamW	0.2
9	ResNet50	AdamW	0.5
10	VGG19	Adam	No Dropout
11	VGG19	Adam	0.2
12	VGG19	Adam	0.5
13	VGG19	Nadam	No Dropout
14	VGG19	Nadam	0.2

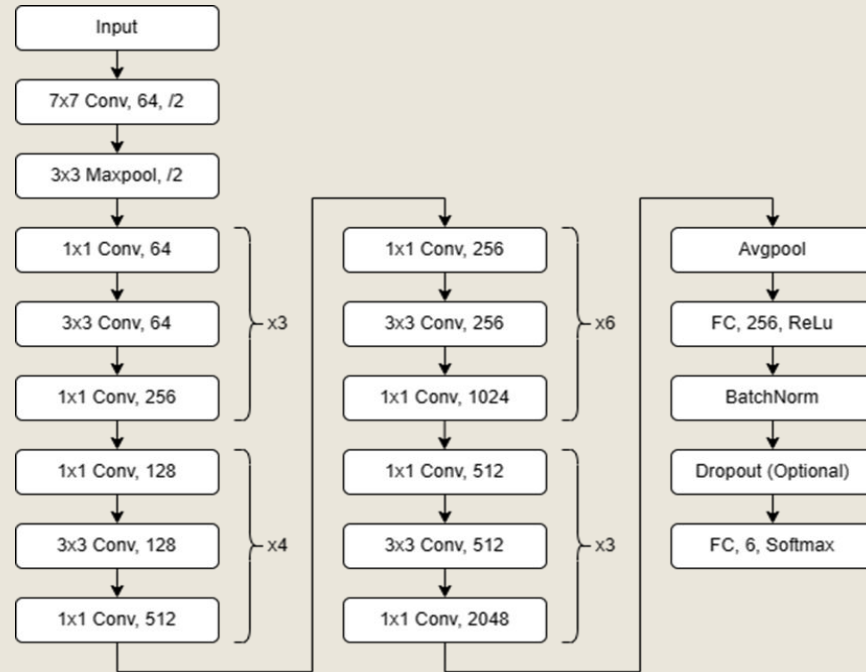
Skenario	Pretrained Model	Optimizers	Dropout Layer
15	VGG19	Nadam	0.5
16	VGG19	AdamW	No Dropout
17	VGG19	AdamW	0.2
18	VGG19	AdamW	0.5
19	DenseNet121	Adam	No Dropout
20	DenseNet121	Adam	0.2
21	DenseNet121	Adam	0.5
22	DenseNet121	Nadam	No Dropout
23	DenseNet121	Nadam	0.2
24	DenseNet121	Nadam	0.5
25	DenseNet121	AdamW	No Dropout
26	DenseNet121	AdamW	0.2
27	DenseNet121	AdamW	0.5

SKENARIO PELATIHAN MODEL

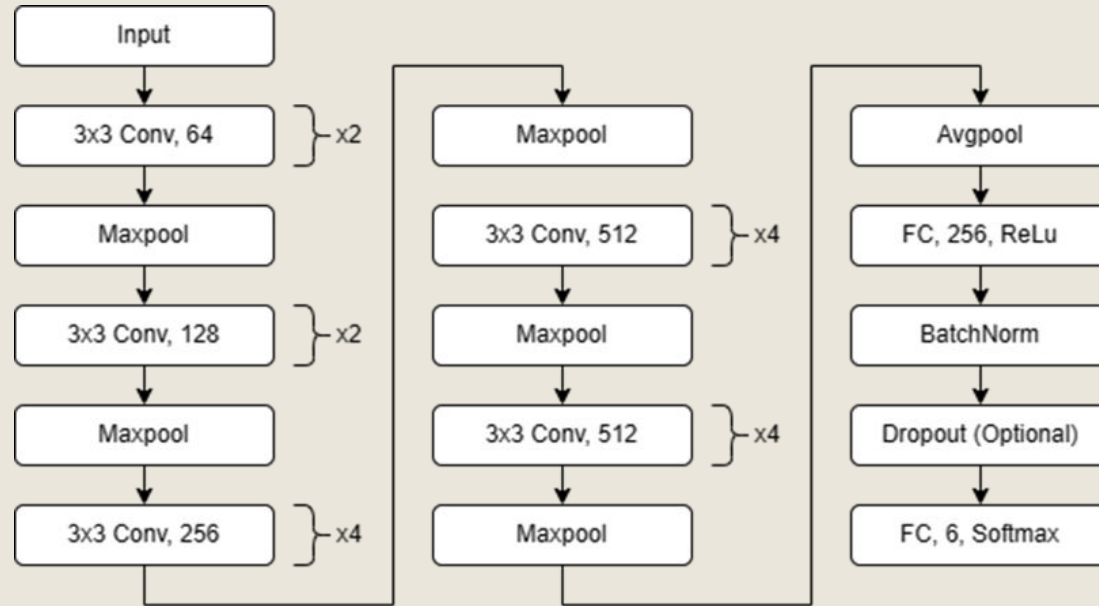
- Pelatihan model hanya mengambil bobot dari *pretrained model* dan menggunakan *layer classifier* seperti gambar disamping
- Perbandingan *Dropout* digunakan pada *layer classifier* sesuai skenario
- Ukuran *input* disesuaikan dengan ukuran *input* pada *pretrained model* yaitu 224x224 pixel



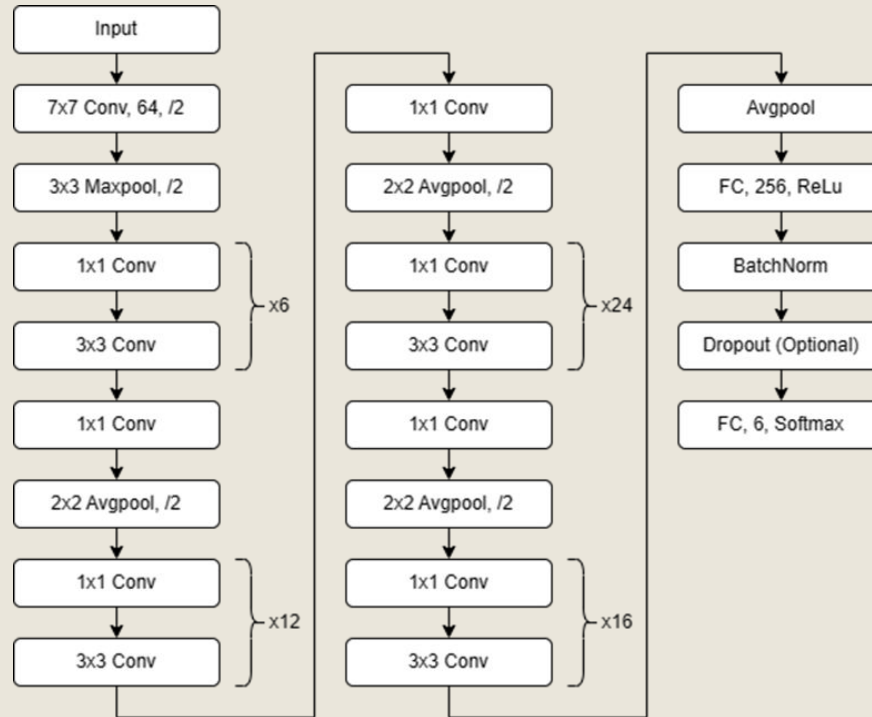
ARSITEKTUR MODEL RESNET50



ARSITEKTUR MODEL VGG19



ARSITEKTUR MODEL DENSENET121



PELATIHAN MODEL

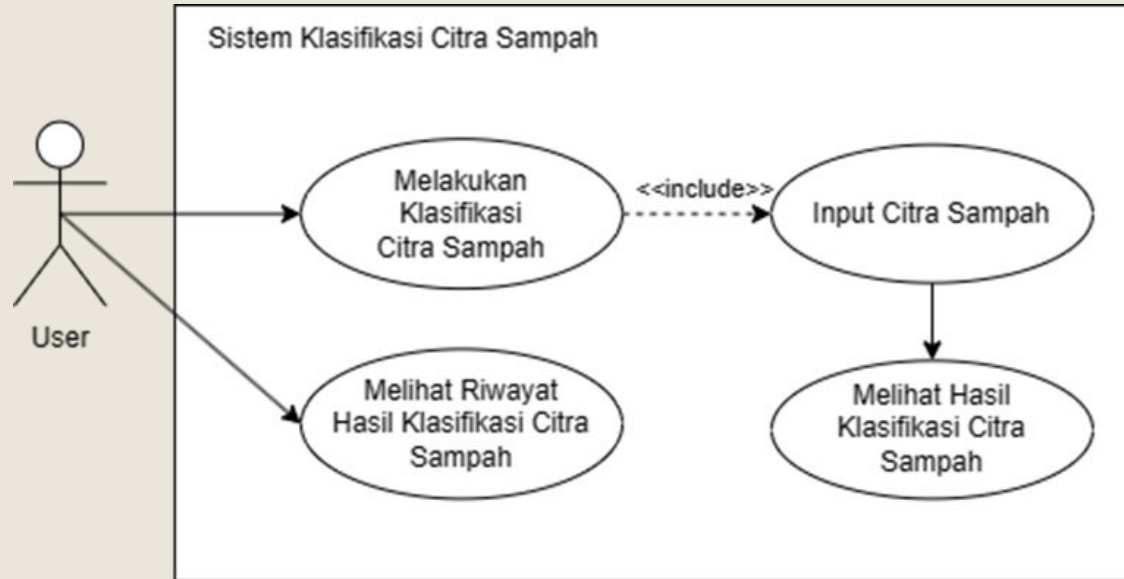
- Pelatihan model menggunakan *pretrained model* sesuai dengan skenario
- Perbandingan *Optimizers* digunakan pada pelatihan model yang berbeda untuk setiap skenario
- *Hyperparameter* model :
 - *Learning Rate* : 0.0001
 - *Epochs* : 10 per *fold*

EVALUASI MODEL

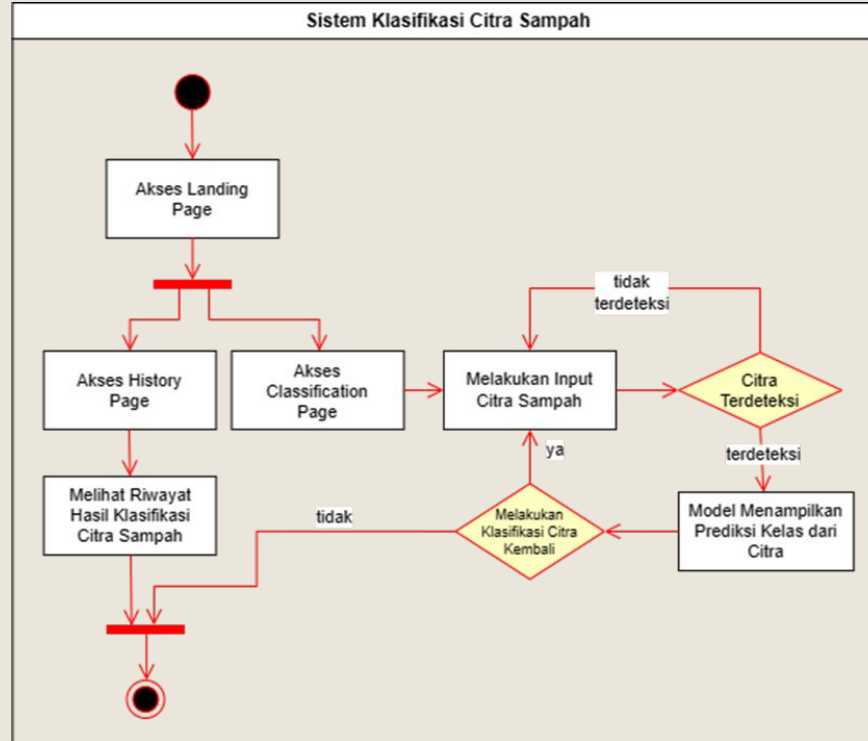
- Evaluasi model dilakukan dengan membuat *Confusion Matrix* untuk jumlah keseluruhan dari setiap *fold* untuk setiap skenario
- *Confusion Matrix* disimpan dalam bentuk *image*
- Nilai *True Positive*, *False Positive*, dan *False Negative* dihitung untuk setiap kelasnya dan digunakan untuk menghitung nilai *Precision*, *Recall*, dan *F1 Score* dari setiap skenario pelatihan

Actual class		Assigned class	
		Positive	Negative
	Positive	TP	FN
	Negative	FP	TN

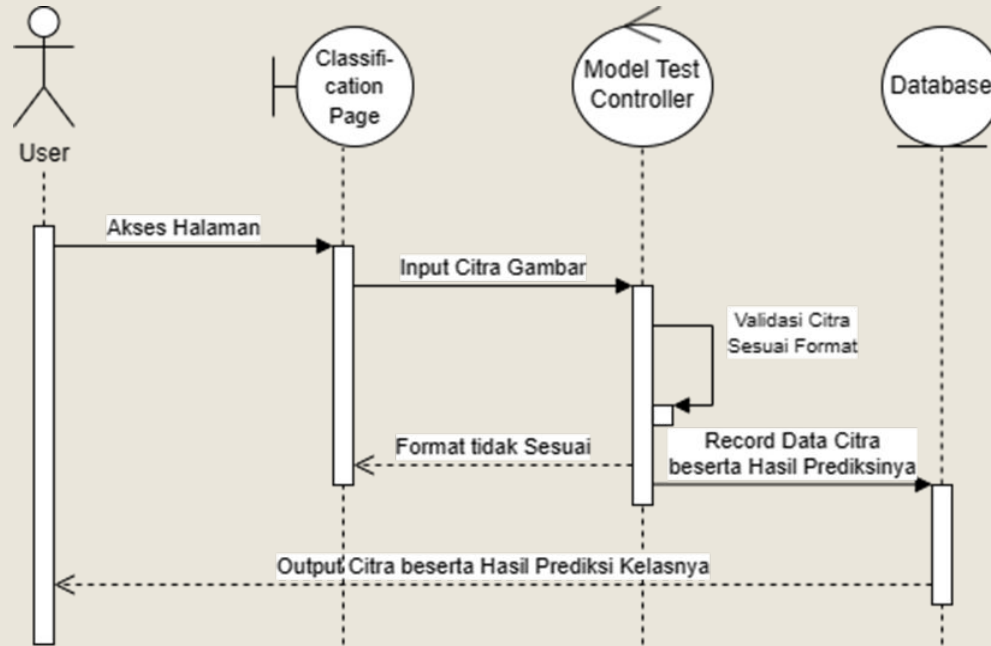
PERANCANGAN APLIKASI - *USE CASE DIAGRAM*



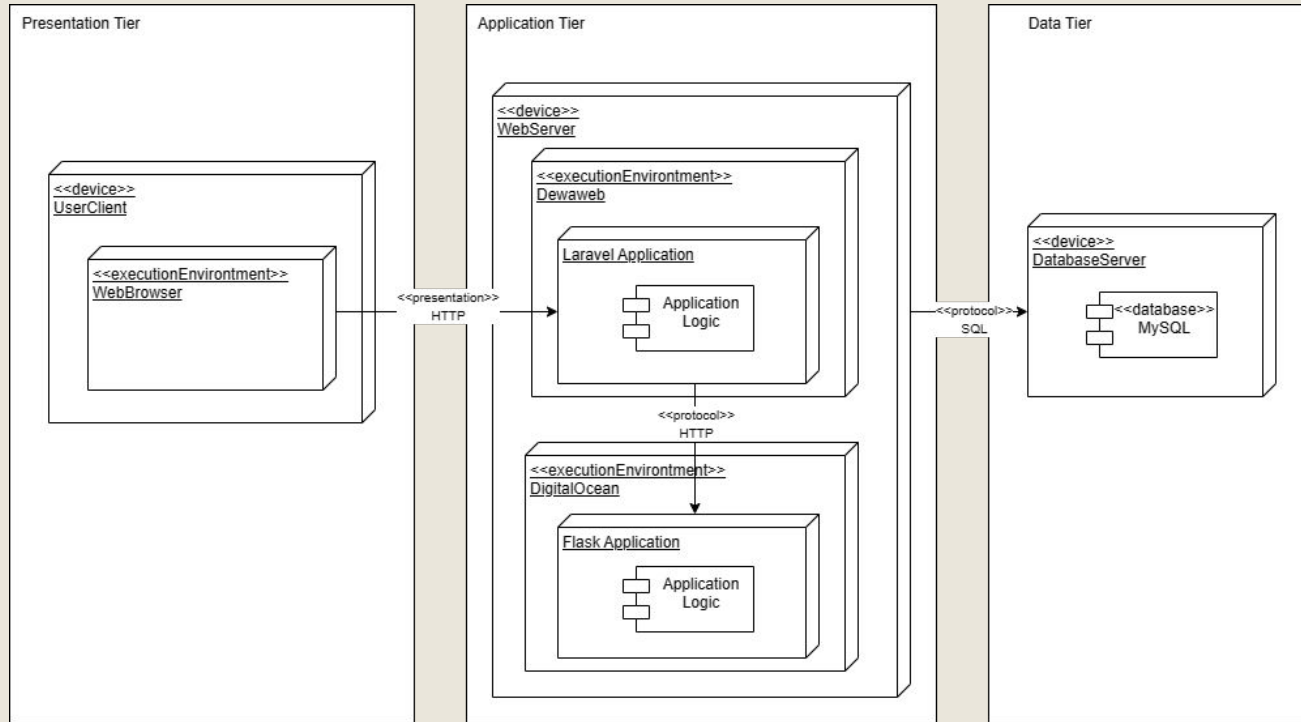
PERANCANGAN APLIKASI - *ACTIVITY DIAGRAM*



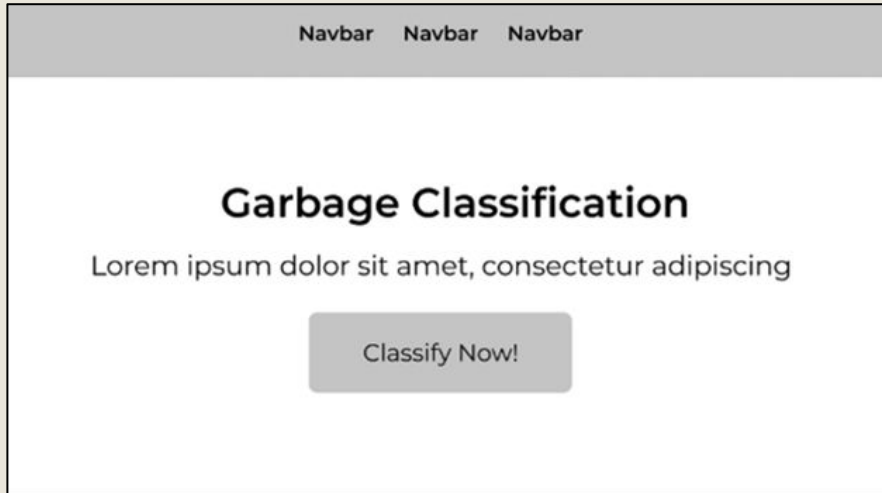
PERANCANGAN APLIKASI - *SEQUENCE DIAGRAM*





PERANCANGAN APLIKASI - *DEPLOYMENT DIAGRAM*



WIREFRAME - LANDING PAGE & HISTORY PAGE



Classification History

No	Image	Prediction	Timestamp
1		Lorem Ipsum	Lorem Ipsum
2		Lorem Ipsum	Lorem Ipsum

The wireframe for the 'Classification History' page includes a title and a table with four columns: 'No', 'Image', 'Prediction', and 'Timestamp'. The table contains two rows of data, each with a placeholder image represented by a square with an 'X' inside.


WIREFRAME - CLASSIFICATION PAGE & PREDICTION PAGE

Classify Garbage

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore

No file chosen

Classification Result



Prediction : Lorem Ipsum

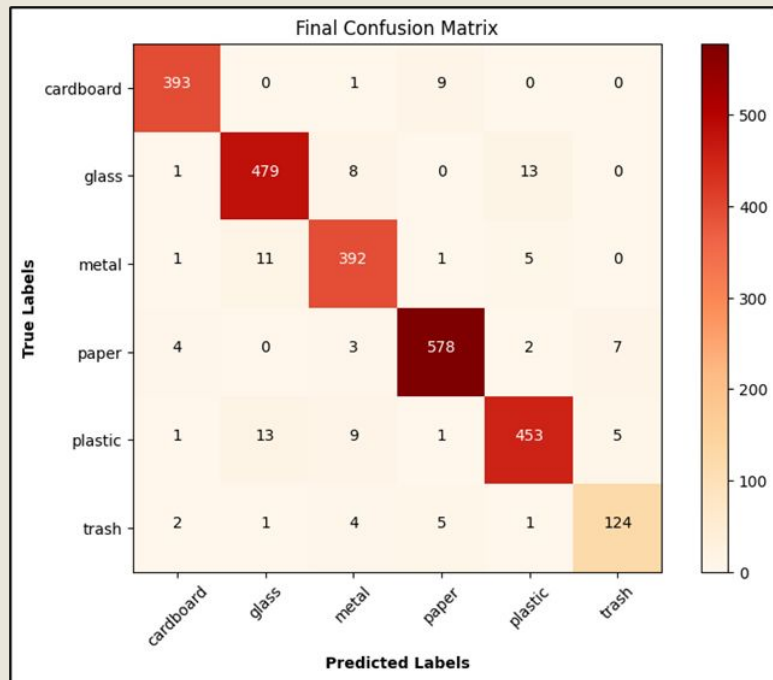
HASIL PELATIHAN MODEL PER SKENARIO

Skenario	Hyperparameters	Precision	Recall	F1 Score
1	ResNet50; Adam; No Dropout	0.9514	0.9509	0.9511
2	ResNet50; Adam; Dropout(0.2)	0.9470	0.9444	0.9456
3	ResNet50; Adam; Dropout(0.5)	0.9233	0.9170	0.9200
4	ResNet50; Nadam; No Dropout	0.9493	0.9468	0.9479
5	ResNet50; Nadam; Dropout(0.2)	0.9334	0.9339	0.9336
6	ResNet50; Nadam; Dropout(0.5)	0.9308	0.9235	0.9270
7	ResNet50; AdamW; No Dropout	0.9436	0.9372	0.9401
8	ResNet50; AdamW; Dropout(0.2)	0.9461	0.9443	0.9452
9	ResNet50; AdamW; Dropout(0.5)	0.9287	0.9204	0.9243
10	VGG19; Adam; No Dropout	0.8858	0.8684	0.8759
11	VGG19; Adam; Dropout(0.2)	0.8756	0.8549	0.8635
12	VGG19; Adam; Dropout(0.5)	0.8496	0.8208	0.8319
13	VGG19; Nadam; No Dropout	0.8681	0.8524	0.8592
14	VGG19; Nadam; Dropout(0.2)	0.8723	0.8505	0.8589

Skenario	Hyperparameters	Precision	Recall	F1 Score
15	VGG19; Nadam; Dropout(0.5)	0.8529	0.8352	0.8427
16	VGG19; AdamW; No Dropout	0.8654	0.8561	0.8604
17	VGG19; AdamW; Dropout(0.2)	0.8787	0.8570	0.8658
18	VGG19; AdamW; Dropout(0.5)	0.8497	0.8335	0.8401
19	DenseNet121; Adam; No Dropout	0.9055	0.9042	0.9048
20	DenseNet121; Adam; Dropout(0.2)	0.9063	0.9066	0.9061
21	DenseNet121; Adam; Dropout(0.5)	0.8797	0.8691	0.8736
22	DenseNet121; Nadam; No Dropout	0.9122	0.9062	0.9090
23	DenseNet121; Nadam; Dropout(0.2)	0.9067	0.8967	0.9011
24	DenseNet121; Nadam; Dropout(0.5)	0.8857	0.8696	0.8761
25	DenseNet121; AdamW; No Dropout	0.9109	0.9006	0.9053
26	DenseNet121; AdamW; Dropout(0.2)	0.9040	0.8906	0.8966
27	DenseNet121; AdamW; Dropout(0.5)	0.8929	0.8874	0.8897

Skenario dengan *F1 Score* paling optimal adalah skenario 1 dengan perbandingan *hyperparameter pretrained model* ResNet50, *Optimizer* Adam, dan tidak menggunakan *Dropout* dengan *F1 Score* 0.9511

CONFUSION MATRIX



1. Cardboard

TP : 393

FP : 9

FN : 10

2. Glass

TP : 479

FP : 25

FN : 22

3. Metal

TP : 392

FP : 25

FN : 18

4. Paper

TP : 578

FP : 16

FN : 16

5. Plastic

TP : 453

FP : 21

FN : 29

6. Trash

TP : 124

FP : 12

FN : 13

PERHITUNGAN *PRECISION* DAN *RECALL*

$$Precision = \frac{TP}{TP + FP}$$

Class	Perhitungan
Cardboard	$\frac{393}{393 + (1 + 1 + 4 + 1 + 2)} = 0.9776$
Glass	$\frac{479}{479 + (11 + 13 + 1)} = 0.9504$
Metal	$\frac{392}{392 + (1 + 8 + 3 + 9 + 4)} = 0.9400$
Paper	$\frac{578}{578 + (9 + 1 + 1 + 5)} = 0.9731$
Plastic	$\frac{453}{453 + (13 + 5 + 2 + 1)} = 0.9557$
Trash	$\frac{124}{124 + (7 + 5)} = 0.9118$
	$\frac{0.9776 + 0.9504 + 0.9400 + 0.9731 + 0.9557 + 0.9118}{6} = 0.9514$

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

Class	Perhitungan
Cardboard	$\frac{393}{393 + (1 + 9)} = 0.9752$
Glass	$\frac{479}{479 + (1 + 8 + 13)} = 0.9561$
Metal	$\frac{392}{392 + (1 + 11 + 1 + 5)} = 0.9561$
Paper	$\frac{578}{578 + (4 + 3 + 2 + 7)} = 0.9731$
Plastic	$\frac{453}{453 + (1 + 13 + 9 + 1 + 5)} = 0.9398$
Trash	$\frac{124}{124 + (2 + 1 + 4 + 5 + 1)} = 0.9051$
	$\frac{0.9752 + 0.9561 + 0.9561 + 0.9731 + 0.9398 + 0.9051}{6} = 0.9509$

PERHITUNGAN *F1 SCORE*

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Class	Perhitungan
Cardboard	$2 \times \frac{0.9776 \times 0.9752}{0.9776 + 0.9752} = 0.9764$
Glass	$2 \times \frac{0.9504 \times 0.9561}{0.9504 + 0.9561} = 0.9532$
Metal	$2 \times \frac{0.9400 \times 0.9561}{0.9400 + 0.9561} = 0.9480$
Paper	$2 \times \frac{0.9731 \times 0.9731}{0.9731 + 0.9731} = 0.9731$
Plastic	$2 \times \frac{0.9557 \times 0.9398}{0.9557 + 0.9398} = 0.9477$
Trash	$2 \times \frac{0.9118 \times 0.9051}{0.9118 + 0.9051} = 0.9084$
	$\frac{0.9764 + 0.9532 + 0.9480 + 0.9731 + 0.9477 + 0.9084}{6} = 0.9511$

SAMPEL KESALAHAN KLASIFIKASI



Plastic -> Glass



Glass -> Plastic



Glass -> Metal



Metal -> Glass



Metal -> Plastic



Plastic -> Metal



Paper -> Cardboard



Cardboard -> Paper

ANALISIS HASIL PERBANDINGAN *PRETRAINED MODEL*

- Penggunaan *pretrained model* yang kompleks seperti DenseNet121 dengan koneksi antar *layer* yang padat menghasilkan *F1 Score* yang terbaik sebesar 0.9090. Hal ini menandakan model mengalami kesulitan dalam generalisasi saat melakukan klasifikasi citra
- Penggunaan *pretrained model* dengan arsitektur tradisional seperti VGG19 menghasilkan *F1 Score* yang terbaik sebesar 0.8759. Hal ini menandakan performa model kurang baik dalam melakukan klasifikasi citra
- Penggunaan *pretrained model* seperti ResNet50 dengan koneksi sederhana antar *output layer* berhasil menghasilkan performa yang optimal untuk klasifikasi citra sampah pada *dataset* Garbage Classification

ANALISIS HASIL PERBANDINGAN *OPTIMIZERS*

- Penggunaan *Optimizers* Adam menunjukkan performa yang lebih baik pada *pretrained model* ResNet50 dan VGG19, sedangkan Nadam lebih baik pada *pretrained model* DenseNet121
- Penggunaan *Optimizers* yang lebih kompleks seperti Nadam dengan implementasi *Nesterov's Accelerated Gradient* atau AdamW dengan implementasi *Weight Decay* tidak selalu memberikan performa yang lebih baik dibandingkan *Optimizer* yang lebih sederhana seperti Adam dalam klasifikasi citra sampah dengan *dataset* Garbage Classification

ANALISIS HASIL PERBANDINGAN *DROPOUT*

- Penggunaan *Dropout* sebesar 0.2 mayoritas tidak menghasilkan performa yang lebih baik dibandingkan tidak menggunakan *Dropout*.
- Penggunaan *Dropout* sebesar 0.5 sudah jelas tidak menghasilkan performa yang lebih baik
- Dengan tidak menggunakan *Dropout layer* menghasilkan performa yang lebih baik dibandingkan menggunakan *Dropout layer*. Hal ini menunjukkan *feature map* dari data pada *dataset* Garbage Classification mudah untuk diekstraksi

IMPLEMENTASI PEMBUATAN APLIKASI

- Aplikasi berbasis *website* dibuat dengan *framework* Laravel untuk memudahkan pembuatan aplikasi dengan memanfaatkan fitur *Model-View-Controller* (MVC)
- Pada *database*, *table Images* dibuat untuk menyimpan *path* dari citra yang disimpan pada folder aplikasi, hasil prediksi, dan probabilitasnya
- Pada halaman *classification page*, *form request* POST dibuat untuk *menginput* citra. Data *request* akan diproses pada *JavaScript* untuk dikirimkan ke *endpoint* Flask untuk diprediksi. Kemudian citra beserta hasil prediksi dan probabilitasnya akan dikirimkan ke *controller* untuk disimpan dalam *database*.

IMPLEMENTASI FLASK PADA APLIKASI

- Flask API digunakan dengan mendefinisikan *endpoint* untuk melakukan prediksi citra
- Model yang sudah *compile* menjadi format h5 dimasukkan kedalam Flask
- Citra yang *diinputkan* pada *website* menggunakan *request POST* dan *diresize* terlebih dahulu menjadi 224x224 *pixel*. Kemudian citra *dipredict* oleh model dan menghasilkan *output* prediksi *class* dari citra beserta probabilitasnya

IMPLEMENTASI APLIKASI



garbage-classification.my.id

KESIMPULAN

- Pembuatan model *Convolutional Neural Network* untuk klasifikasi citra sampah dapat dilakukan dengan menggunakan *framework* TensorFlow dan *library* dari Keras dengan bahasa pemrograman Python
- Konfigurasi *hyperparameter* model *Convolutional Neural Network* yang menghasilkan performa paling optimal yaitu menggunakan arsitektur ResNet50, menggunakan *Optimizer* Adam, dan tidak menggunakan *Dropout layer* yang menghasilkan *F1 Score* sebesar 0.9511
- Penerapan model *Convolutional Neural Network* pada aplikasi klasifikasi citra sampah berbasis *website* dapat dilakukan dengan menggunakan Flask API untuk melakukan prediksi pada citra

SARAN

- *Dataset* yang digunakan memiliki data untuk setiap kelas yang hanya sedikit dan terdapat jumlah data yang *imbalance* untuk kelas *Trash*. Oleh karena itu, diperlukan jumlah data yang lebih banyak sehingga model *Convolutional Neural Network* dapat menghasilkan performa yang lebih optimal
- Diperlukan percobaan implementasi model dengan metode seperti *Oversampling* atau penambahan *data augmentation* untuk mengatasi data yang *imbalance*
- Diperlukan perangkat lunak dan perangkat keras yang memiliki performa lebih baik untuk mempercepat dan mempermudah proses *training* model sehingga penelitian dapat berjalan dengan efektif

THANK YOU