SKRIPSI

IMPLEMENTASI *CONVOLUTIONAL NEURAL NETWORK* UNTUK DETEKSI CACAT KAIN

Disusun dan diajukan oleh:

NURFAIDAH
D121181339

PROGRAM STUDI SARJANA TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS HASANUDDIN
GOWA
2024

LEMBAR PENGESAHAN SKRIPSI

IMPLEMENTASI *CONVOLUTIONAL NEURAL NETWORK* UNTUK DETEKSI CACAT KAIN

Disusun dan diajukan oleh

Nurfaidah
D121181339

Telah dipertahankan di hadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian Studi Program Sarjana Program Studi Teknik Informatika
Fakultas Teknik Universitas Hasanuddin
Pada tanggal 12 Februari 2024
dan dinyatakan telah memenuhi syarat kelulusan

Menyetujui,

Pembimbing Utama,	Pembimbing Pendamping,
Dr. Ir. Ingrid Nurtanio, M.T.	Prof. Dr. Ir. Syafruddin Syarif, M.T.
NIP 19610813 198811 2 001	NIP 19611125 198802 1 001

Ketua Program Studi,

Prof. Dr. Ir. Indrabayu, S.T., M.T., M.Bus.Sys., IPM, ASEAN. Eng.
NIP 19750716 200212 1 004

PERNYATAAN KEASLIAN

Yang bertanda tangan dibawah ini ;

Nama	:	Nurfaidah
NIM	:	D121181339
Program Studi	:	Teknik Informatika
Jenjang	:	S1

Menyatakan dengan ini bahwa karya tulisan saya berjudul

{Implementasi *Convolutional Neural Network* untuk Deteksi Cacat Kain}

Adalah karya tulisan saya sendiri dan bukan merupakan pengambilan alihan tulisan orang lain dan bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri.

Semua informasi yang ditulis dalam skripsi yang berasal dari penulis lain telah diberi penghargaan, yakni dengan mengutip sumber dan tahun penerbitannya. Oleh karena itu semua tulisan dalam skripsi ini sepenuhnya menjadi tanggung jawab penulis. Apabila ada pihak manapun yang merasa ada kesamaan judul dan atau hasil temuan dalam skripsi ini, maka penulis siap untuk diklarifikasi dan mempertanggungjawabkan segala resiko.

Segala data dan informasi yang diperoleh selama proses pembuatan skripsi, yang akan dipublikasi oleh Penulis di masa depan harus mendapat persetujuan dari Dosen Pembimbing.

Apabila dikemudian hari terbukti atau dapat dibuktikan bahwa sebagian atau keseluruhan isi skripsi ini hasil karya orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Gowa, 12 Februari 2024

Yang Menyatakan

Nurfaidah

ABSTRAK

NURFAIDAH. *Implementasi Convolutional Neural Network untuk Deteksi Cacat Kain* (dibimbing oleh Ingrid Nurtanio dan Syafruddin Syarif)

Kain adalah bahan yang terbuat dari hasil tenunan benang yang dapat diolah menjadi pakaian, dan berbagai macam kebutuhan sandang lainnya. Dalam proses pembelian kain ada kalanya kain yang dibeli mengalami cacat. Cacat tersebut ada yang mudah dilihat ada juga yang samar. *Factory outlet* adalah toko ritel yang menjual barang-barang hasil limpahan dari pabrik (khususnya barang sandang) yang merupakan sisa kelebihan produksi untuk keperluan ekspor atau disisihkan oleh pabrik dikarenakan terdapat cacat pada produk tersebut. Masalah muncul ketika konsumen tidak dapat dengan mudah mendeteksi cacat pada produk yang ingin dibeli, baik kain di pasaran maupun barang sandang di *factory outlet*. Hal tersebut mengakibatkan pembelian produk menjadi tidak memenuhi standar kualitas yang diharapkan oleh konsumen. Oleh karena itu, diperlukan suatu solusi yang dapat membantu masyarakat dalam mendeteksi kecacatan pada kain sebelum melakukan pembelian. Sistem deteksi cacat kain pada penelitian ini dibangun menggunakan metode *Convolutional Neural Network*, arsitektur MobileNetV2 dengan *hyperparameter tunning* yang menghasilkan model lebih optimal yaitu *epoch* 30, *batch size* 10, *steps per epoch* dan *validation steps* diatur *default*, serta *learning rate* 0.0001. Dari proses pelatihan dengan jumlah data latih 405 citra dan proses validasi dengan jumlah data validasi 114 citra, memberikan hasil akurasi *training* 0.9855 dan *loss* 0.7092, serta hasil akurasi *validation* 0.9912 dan *loss* 0.7024. Kemudian tingkat keberhasilan sistem deteksi cacat kain yang diuji pada aplikasi *mobile* (android) dengan jumlah data uji 57 citra yaitu akurasi sebesar 96,49%. Hal ini menujukkan bahwa sistem mampu mendeteksi cacat kain dengan baik.

Kata Kunci: *Convolutional Neural Network*, MobileNetV2, Deteksi Cacat Kain, Android

ABSTRACT

NURFAIDAH. *Implementation of Convolutional Neural Network for Fabric Defect Detection* (supervised by Ingrid Nurtanio and Syafruddin Syarif)

Fabric is a material made from the weaving of threads that can be processed into clothing and various other clothing needs. In the process of purchasing fabric, there are times when the purchased fabric is defective. Some defects are easily visible, while others are subtle. A factory outlet is a retail store that sells goods produced in excess from factories, particularly clothing items that are surplus due to overproduction for export purposes or are rejected by the factory due to defects. The problem arises when consumers cannot easily detect defects in the products they want to purchase, whether it's fabric in the market or clothing items in a factory outlet. This results in the purchased products not meeting the quality standards expected by consumers. Therefore, a solution is needed to help the public detect defects in fabric before making a purchase. The fabric defect detection system in this research was built using the Convolutional Neural Network method, MobileNetV2 architecture with hyperparameter tuning which produces a more optimal model, namely epoch 30, batch size 10, steps per epoch and validation steps set by default, and a learning rate of 0.0001. From the training process with a total of 405 images of training data and a validation process with a total of 114 images of validation data, the training accuracy results were 0.9855 and the loss was 0.7092, as well as the validation accuracy results were 0.9912 and the loss was 0.7024. Then the success rate of the fabric defect detection system which was tested on a mobile application (Android) with a total of 57 image test data was an accuracy of 96.49%. This shows that the system is able to detect fabric defects well.

Keywords: Convolutional Neural Network, MobileNetV2, Fabric Defect Detection, Android

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR SINGKATAN DAN ARTI SIMBOL

Lambang/Singkatan	Arti dan Keterangan
AIMP	*Artificial Intelligence and Multimedia Processing*
BN	*Batch Normalization*
CPU	*Central Processing Unit*
CNN	*Convolutional Neural Network*
DSC	*Depthwise Separable Convolution*
GPU	*Graphics Processing Unit*
ML	*Machine Learning*
TF	TensorFlow
TPU	*Tensor Processing Unit*
RGB	*Red, Green, Blue*

DAFTAR LAMPIRAN

KATA PENGANTAR

Bismillahirrahmanirrahim.
Assalamu'alaikum Warahmatullahi Wabarakatuh.

Puji syukur kehadirat Allah *Subhanahu wata'ala* atas rahmat dan karunia-Nya sehingga penelitian dan penyusunan skripsi dengan judul "**Implementasi *Convolutional Neural Network* untuk Deteksi Cacat Kain**" dapat diselesaikan sebagai salah satu syarat dalam menyelesaikan jenjang Strata-1 pada Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin.

Masa perkuliahan hingga penyelesaian penulisan skripsi ini merupakan perjuangan yang panjang bagi penulis. Banyak hambatan yang dialami, namun dengan izin Allah *Subhanahu wata'ala* serta doa dan bantuan dari berbagai pihak sehingga dapat terselesaikan dengan baik. Oleh sebab itu, dengan penuh kerendahan hati penulis menyampaikan terima kasih kepada:

1. Kedua orang tua penulis, Bapak Nuhung dan Ibu Nurbia yang telah memberikan kasih sayang, kesempatan dan kepercayaan serta dukungan moral dan materil kepada penulis selama menempuh pendidikan di Universitas Hasanuddin. Kepada saudaraku Nur Rahmi Nuhung, M.Pd., Mursyidah, S.M., Muhammad Ma'ruf dan Muhammad Syarif terima kasih atas motivasi dan dukungan yang tiada henti hingga penulis mampu mencapai titik ini;
2. Bapak Prof. Dr. Ir. Indrabayu, S.T., M.T., M.Bus.Sys., IPM, ASEAN. Eng., selaku Ketua Departemen Teknik Informatika atas inspirasi dan bimbingan selama perkuliahan;
3. Ibu Dr. Ir. Ingrid Nurtanio, M.T., selaku pembimbing utama dan Bapak Prof. Dr. Ir. Syafruddin Syarif, M.T., selaku pembimbing pendamping yang telah meluangkan waktu dalam memberikan bimbingan, saran dan motivasi yang sangat berarti sejak penyusunan proposal hingga terselesainya penulisan skripsi ini;
4. Bapak A. Ais Prayogi Alimuddin, S.T., M.Eng. dan Bapak Muhammad Alief Fahdal Imran Oemar, ST., M.Sc., selaku tim penguji;
5. Bapak Dr. Eng. Zulkifli Tahir, S.T., M.Sc., selaku dosen pembimbing akademik yang senantiasa memberikan bimbingan dan motivasi selama perkuliahan;
6. Segenap Dosen Pengampu Mata Kuliah yang telah mengajarkan ilmu selama penulis menempuh Pendidikan;
7. Segenap Staf Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin yang telah membantu kelancaran administrasi penyelesaian tugas akhir penulis;
8. Synchronous18 selaku teman seperjuangan atas segala bantuan dan momen bersama yang turut mewarnai masa perkuliahan; dan
9. Semua pihak yang tidak sempat penulis sebutkan satu persatu yang telah membantu penulis selama penyelesaian skripsi ini.

Penulis berharap, semoga segala bentuk kebaikan dari semua pihak akan mendapatkan nilai ibadah serta rida Allah *Subhanahu wata'ala*.

Sebagai penutup, diharapkan kepada pembaca untuk memberikan kritik dan saran atas ketidaksempurnaan skripsi ini. Semoga skripsi ini dapat memberikan manfaat bagi pengembangan ilmu selanjutnya.
Wassalamu'alaikum Warahmatullahi Wabarakatuh.

Gowa, 6 November 2023

Nurfaidah

BAB I
PENDAHULUAN

1.1 Latar Belakang

Menurut pengertian Kamus Besar Bahasa Indonesia (KBBI), (1) kain adalah barang yang ditenun dari benang kapas; (2) kain adalah barang tenunan untuk pakaian atau untuk maksud lain (KBBI, 2023). Kain dijual secara bebas di pasaran, umumnya berupa gulungan (*roll*) atau potongan-potongan dalam ukuran terentu. Dalam proses pembelian kain ada kalanya kain yang dibeli mengalami cacat. Cacat tersebut ada yang mudah dilihat ada juga yang samar (Wicaksono, T. A., & Adler, J., 2018). Akibat gagalnya deteksi cacat kain sebelum dibeli atau diolah menjadi suatu produk dapat menimbulkan kerugian seperti waktu menjadi kurang efisien, penurunan harga produk yang dihasilkan atau kain menjadi tidak dapat digunakan.

Kain dapat diolah menjadi pakaian, alas kaki, selimut, taplak meja dan berbagai macam kebutuhan sandang lainnya. *Factory outlet* adalah toko ritel yang menjual barang-barang hasil limpahan dari pabrik (khususnya barang sandang) yang merupakan sisa kelebihan produksi untuk keperluan ekspor atau disisihkan oleh pabrik dikarenakan terdapat cacat pada produk tersebut. *Factory outlet* umumnya menjual barang-barang merek terkenal dengan harga terjangkau.

Masalah muncul ketika konsumen tidak dapat dengan mudah mendeteksi cacat pada produk yang ingin dibeli, baik kain di pasaran maupun barang sandang di *factory outlet*. Hal tersebut mengakibatkan pembelian produk menjadi tidak memenuhi standar kualitas yang diharapkan oleh konsumen. Oleh karena itu, diperlukan suatu solusi yang dapat membantu masyarakat dalam mendeteksi kecacatan pada kain sebelum melakukan pembelian.

Deteksi cacat kain pernah dilakukan dengan menggunakan berbagai metode, seperti GLCM dan SVM (Wicaksono, T. A., & Adler, J., 2018), *Saliency Histogram Features* dan SVM (Li, M., et al., 2019), YOLO-LFD (Liu, Z., et al., 2019) dan PETM-CNN (Wen, Z., et al., 2020). *Convolutional Neural Network* (CNN) merupakan salah satu metode yang dapat digunakan dalam proses pendeteksian suatu citra. CNN berfungsi sebagai pengganti penglihatan manusia

(Alamy, U. G., 2022). Penggunan metode CNN pernah dilakukan untuk klasifikasi dan deteksi kecacatan pada kemasan kaleng (Kusumawardani, R., & Karningsih, P. D., 2021), *flash* (Alamy, U. G., 2022) dan pengelasan (Deng, H., et al., 2021), dengan tingkat akurasi tertinggi masing-masing adalah 95,56%, 91,03% dan 98,75%, sehingga CNN dinilai dapat mengklasifikasikan citra dan deteksi kecatatan dengan baik.

Berdasarkan uraian di atas, maka penulis mengusulkan judul penelitian "Implementasi *Convolutional Neural Network* untuk Deteksi Cacat Kain".

1.2 Rumusan Masalah

1. Bagaimana cara deteksi cacat kain menggunakan metode *Convolutional Neural Network*?
2. Bagaimana tingkat keberhasilan deteksi kain lubang, noda dan normal menggunakan metode *Convolutional Neural Network*?

1.3 Tujuan Penelitian

Tujuan dari penelitian ini adalah:

1. Untuk membuat sistem yang mampu mendeteksi cacat kain sebagai upaya pengendalian kualitas produk.
2. Untuk mengetahui tingkat keberhasikan sistem deteksi cacat kain menggunakan metode *Convolutional Neural Network.*

1.4 Manfaat Penelitian

Penelitian ini diharapkan memberikan manfaat, antara lain:

1. Memberikan kemudahan bagi masyarakat dalam mengidentifikasi cacat pada kain untuk menjamin kualitas produk.
2. Menambah wawasan mengenai implementasi *Convolutional Neural Network* untuk mendeteksi cacat kain sehingga dapat dijadikan acuan referensi untuk penelitian selanjutnya.

1.5 Ruang Lingkup

1. Data yang digunakan sebagai *dataset* diambil dari repositori data Kaggle sebagai data sekunder.
2. Jenis kain yang digunakan yaitu kain polos atau tidak bermotif.
3. Penelitian ini menggunakan metode *Convolutional Neural Network* yaitu arsitektur MobileNetV2.

BAB II
TINJAUAN PUSTAKA

2.1 Kain

Menurut pengertian Kamus Besar Bahasa Indonesia (KBBI), (1) kain adalah barang yang ditenun dari benang kapas; (2) kain adalah barang tenunan untuk pakaian atau untuk maksud lain (KBBI, 2023). Dalam bahasa sederhana, kain adalah bahan yang terbuat dari hasil tenunan benang. Struktur kain dibentuk dari anyaman dua jenis benang yaitu benang lusi (memanjang) dan benang pakan (melebar) dengan ukuran tertentu (Wicaksono, T. A., & Adler, J., 2018).

Ada berbagai jenis kain seperti katun, poliester, rayon, wol, sutra, dan masih banyak lagi. Kain tersebut dapat diolah menjadi pakaian, alas kaki, selimut, taplak meja dan berbagai macam kebutuhan sandang lainnya.

2.1.1 Cacat kain

Cacat kain adalah kelainan yang tampak pada permukaan kain secara fisik (Bagues, T., 2021). Akibat cacat kain ini akan menurunkan kualitas kain itu sendiri. Contoh cacat kain yang sering terjadi adalah lubang dan noda.

- Lubang adalah ketidaksempurnaan dimana satu atau lebih benang rusak cukup parah sehingga menimbulkan lubang pada kain. Lubang biasanya disebabkan oleh terpotong atau sobeknya kain secara tidak sengaja, jarum yang patah atau komponen mekanis yang kasar (Lim. S., 2018).
- Noda didefinisikan sebagai bintik atau bercak dengan warna berbeda. Penyebab noda pada kain dapat disebabkan oleh berbagai hal, seperti kotoran dari lantai pabrik, minyak dari mesin, dan pewarna (Lim, S., 2018).
- Normal merupakan istilah yang menggambarkan kain atau tekstil yang tidak memiliki cacat atau kerusakan dalam kualitas atau penampilannya. Kain ini memiliki kondisi yang sempurna, tanpa lubang, noda atau masalah lainnya yang mengurangi nilai atau estetika kain tersebut.

2.2 *Machine Learning*

Machine learning adalah cabang dari *artificial intelligence*. Kecerdasan buatan memiliki pengertian yang sangat luas tapi secara umum dapat dipahami sebagai komputer dengan kecerdasan layaknya manusia. Istilah *machine learning* pertama kali dipopulerkan oleh Arthur Samuel, seorang ilmuwan komputer yang memelopori kecerdasan buatan pada tahun 1959. Menurut Arthur Samuel, *machine learning* adalah suatu cabang ilmu yang memberi komputer kemampuan untuk belajar tanpa diprogram secara eksplisit. Algoritma *machine learning* mencari pola tertentu dari setiap kumpulan data yang menentukan kekhasan masing-masing untuk kemudian menyimpulkan sebuah aturan. Selanjutnya, aturan ini dapat digunakan untuk melakukan identifikasi dan prediksi bagi data baru yang relevan dengan model yang kita miliki. Contoh implementasi *machine learning* yaitu penanda email spam, rekomendasi video di Youtube, fitur pengenal wajah pada gambar dan lain sebagainya (Setiani, T. D., dkk., 2023).

Machine learning dibagi menjadi 4 kategori berdasarkan karakteristik data dan jenis supervisi yang didapatkan oleh program selama pelatihan (Setiani, T. D., dkk., 2023), yaitu:

1. *Supervised learning* yaitu kategori *machine learning* yang menyertakan solusi yang diinginkan (yang disebut label) dalam proses pembelajarannya. *Dataset* yang digunakan telah memiliki label dan algoritma kemudian mempelajari pola dari pasangan data dan label tersebut. *Supervised learning* dapat dilihat sebagai sebuah mesin/robot yang belajar menjawab pertanyaan sesuai dengan jawaban yang telah disediakan manusia. Algoritma yang termasuk *supervised learning* diantaranya adalah klasifikasi, regresi, *decision tree*, *support vector machine* dan *neural networks*.
2. *Unsupervised learning*, model *unsupervised learning* melakukan proses belajar sendiri untuk melabeli atau mengelompokkan data. *Unsupervised learning* dapat dilihat sebagai robot/mesin yang berusaha belajar menjawab pertanyaan secara mandiri tanpa ada jawaban yang disediakan manusia. Algoritma yang termasuk *unsupervised learning* diantaranya adalah *clustering*, pendeteksian anomali (*anomaly detection*), dan pengurangan dimensi (*dimension reduction*).

3. *Semi-supervised learning* merupakan gabungan dari *supervised learning* dan *unsupervised learning*. Disini *dataset* untuk pelatihan sebagian memiliki label dan sebagian tidak.
4. *Reinforcement learning* dikenal sebagai model yang belajar menggunakan sistem *reward* dan penalti. Menurut Winder, *reinforcement learning* adalah teknik yang mempelajari bagaimana membuat keputusan terbaik, secara berurutan, untuk memaksimalkan ukuran sukses kehidupan nyata. Entitas pembuat keputusan belajar melalui proses trial dan eror.

2.3 *Deep Learning*

Deep learning adalah cabang *machine learning* dengan algoritma jaringan syaraf tiruan yang dapat belajar dan beradaptasi terhadap sejumlah besar data. Algoritma jaringan syaraf tiruan pada *deep learning* terinspirasi dari struktur otak manusia. Algoritma ini memungkinkan mesin untuk melihat pola dari data yang tidak terstruktur atau data yang fiturnya tidak dapat ditentukan secara langsung. Contohnya, data gambar, teks, audio, dan video (Setiani, T. D., dkk., 2023).

Deep learning mengajarkan komputer untuk melakukan hal yang alami bagi manusia: belajar dari pengalaman. *Deep learning* menggunakan jaringan saraf untuk mempelajari representasi fitur yang berguna langsung dari data. Jaringan saraf menggabungkan beberapa lapisan pemrosesan nonlinier, menggunakan elemen sederhana yang beroperasi secara paralel dan terinspirasi oleh sistem saraf biologis. Model *deep learning* dapat mencapai akurasi canggih dalam klasifikasi objek, terkadang melebihi performa tingkat manusia (MathWorks, n.d.-a).

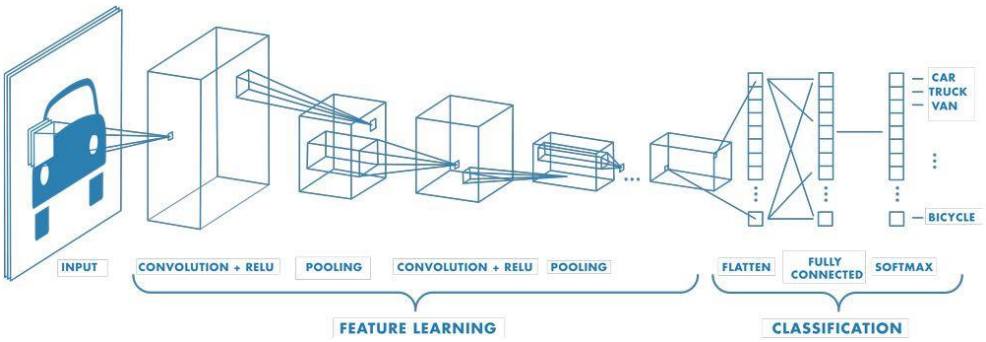
2.4 *Convolutional Neural Network*

Convolutional Neural Network (CNN) adalah arsitektur yang mampu mengenali informasi prediktif suatu objek seperti gambar, teks, potongan suara, dan lain sebagainya (Putra, J. W. G., 2020). CNN merupakan pengembangan dari *multilayer perceptron* (MLP) yang didesain untuk mengolah data dalam bentuk citra. CNN termasuk jenis *deep neural network* karena jaringan yang tinggi dan banyak diaplikasikan pada data citra. Penelitian CNN pertama kali dilakukan oleh

Hubel dan Wiesel (1968) tentang *visual cortex* pada indera penglihatan kucing (Handoyo, A. T., 2021).

2.4.1 Bagaimana CNN bekerja?

Secara garis besar CNN terbagi menjadi dua tahap yaitu *feature learning* dan *classification* yang masing-masing terdiri dari beberapa *layer*. CNN dapat memiliki puluhan atau ratusan *layer* yang masing-masing belajar mendeteksi fitur berbeda pada suatu gambar. Pada Gambar 1 di bawah merupakan contoh jaringan dengan banyak lapisan *convolutional*. Filter diterapkan ke setiap gambar pelatihan pada resolusi berbeda, dan *output* dari setiap gambar yang dikonvolusi digunakan sebagai *input* ke lapisan berikutnya. Filter dapat dimulai dari fitur yang sangat sederhana, seperti kecerahan (*brightness*) dan tepian (*edges*), dan meningkat dalam kompleksitas hingga fitur yang mendefinisikan objek secara unik (MathWorks, n.d.-b).



Gambar 1 Ilustrasi arsitektur *Convolutional Neural Network*
Sumber: MathWorks (n.d.-b)

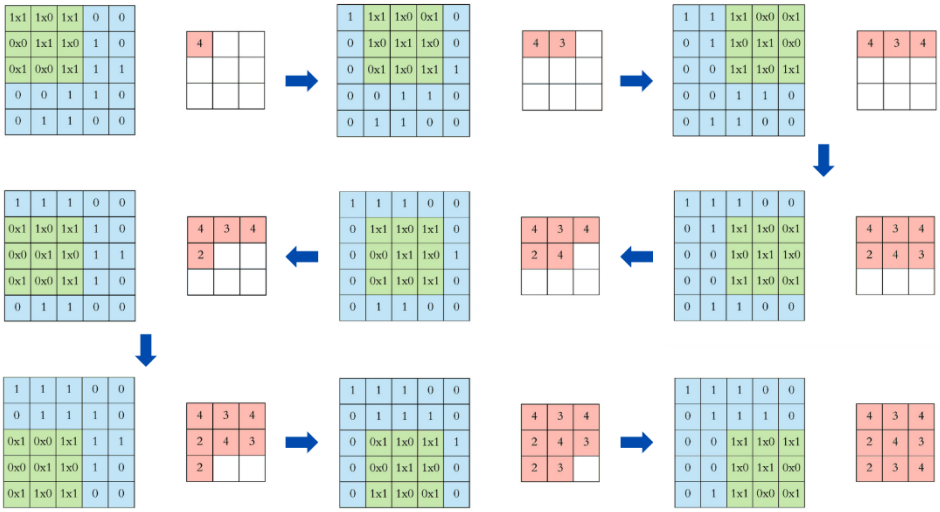
a. *Feature learning*

Proses yang terjadi pada bagian ini adalah melakukan "*encoding*" dari sebuah *image* menjadi *features* yang berupa angka-angka yang merepresentasikan *image* tersebut (*feature extraction*) (Lina, Q., 2019). *Feature learning* ini terdiri dari *convolutional layer* dan *pooling layer*.

- *Convolutional layer* + ReLU

  Pada *convolutional layer* dilakukan pengurangan dimensi citra dengan menggunakan filter untuk melakukan ekstraksi ciri dari citra. Konvolusi dilakukan dengan perkalian antara piksel dari citra dengan ukuran filter. *Output* dari proses konvolusi yaitu

berupa *feature map* yang akan digunakan sebagai *input* pada lapisan berikutnya (Shidiq, A. L. A., dkk., 2022).

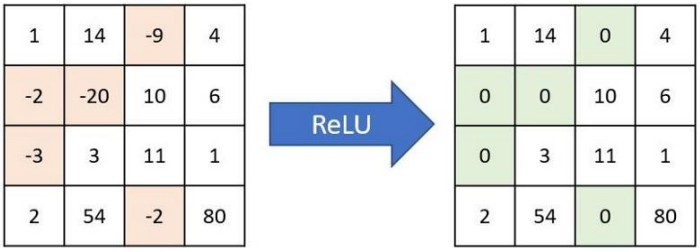


Gambar 2 Ilustrasi operasi *convolution*

Gambar 2 di atas menunjukkan cara kerja operasi *convolution*. Filter ditunjukkan dengan blok berwarna hijau bergerak di atas *input* (ditandai kotak berwarna biru) dan jumlah dari hasil operasi *convolution* masuk ke *feature map* (kotak merah). Untuk memudahkan penjelasan, visualiasi di atas berbentuk 2 dimensi. Tetapi yang sebenarnya terjadi adalah CNN berjalan dalam 3 dimensi. Setiap gambar direpresentasikan sebagai matriks 3D dengan dimensi *width*, *height*, dan *depth*. *Depth* merupakan dimensi karena adanya warna yang digunakan dalam gambar (RGB). Beberapa operasi *convolution* pada data *input* akan dilakukan menggunakan filter yang berbeda. Hasilnya juga akan disimpan dalam *feature map* yang berbeda. Namun pada akhirnya, semua *feature map* ini akan diambil dan digabungkan sebagai hasil akhir dari *convolution layer* (Trivusi, 2022).

Rectified Linear Unit (ReLU) adalah *layer* tambahan yang memungkinkan pelatihan yang lebih cepat dan efektif dengan memetakan nilai negatif ke nol dan mempertahankan nilai positif. Pada dasarnya ReLU adalah operasi per-piksel dengan cara

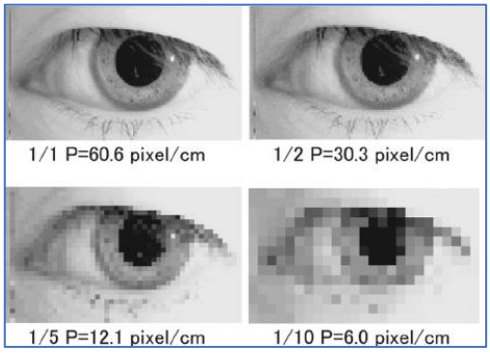
mengganti nilai negatif piksel di dalam *feature map* menjadi nol (Munir, R., 2022). Ilustrasi operasi ReLU dapat dilihat pada Gambar 3 berikut.



Gambar 3 Ilustrasi operasi ReLU
Sumber: Bhatt, A. (2020)

- *Pooling layer*

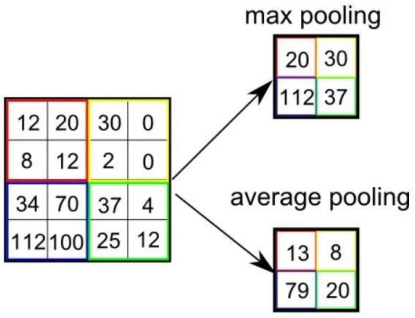
  Umumnya setelah proses konvolusi pada gambar *input*, akan dilakukan proses *pooling*. *Pooling* adalah proses untuk mengurangi resolusi gambar dengan tetap mempertahankan informasi pada gambar (Setiani, T. D., dkk., 2023). Contohnya pada Gambar 4 berikut dimana ketika resolusi dikurangi sampai batas tertentu, namun masih bisa mendapatkan informasi mengenai objek pada gambar tersebut.



Gambar 4 Ilustrasi *pooling* pada citra
Sumber: Setiani, T. D., dkk. (2023)

Pooling layer dilakukan untuk mengurangi dimensi dari *feature map* sehingga mempercepat komputasi karena parameter yang harus di*update* semakin sedikit dan mengatasi *overfitting* (Lina, Q., 2019). Terdapat dua jenis *pooling* yang dapat digunakan, yaitu *max pooling* dan *average pooling*. *Max pooling* akan membagi *output* dalam beberapa *grid*, dan setiap pergeseran

filter akan mengambil nilai terbesar dari setiap *grid*, sedangkan *average pooling* akan mengambil nilai rata-rata (Trivusi, 2022). Adapun contoh proses dari *pooling* dapat dilihat pada Gambar 5 berikut.



Gambar 5 *Max pooling* dan *average pooling*
Sumber: Saha, S. (2018)

b. *Classification*

Setelah deteksi fitur, arsitektur CNN beralih ke klasifikasi (Munir, R., 2022). Klasifikasi terdiri dari *fully-connected layer* (*layer* yang terhubung secara penuh). *Layer* ini hanya dapat menerima data berdimensi 1, sedangkan *feature map* yang dihasilkan dari *feature extraction* masih berbentuk *multidimensional array*, sehingga harus melakukan "*flatten*" atau *reshape feature map* menjadi sebuah vektor agar bisa digunakan sebagai *input* pada *fully-connected layer* (Lina, Q., 2019).

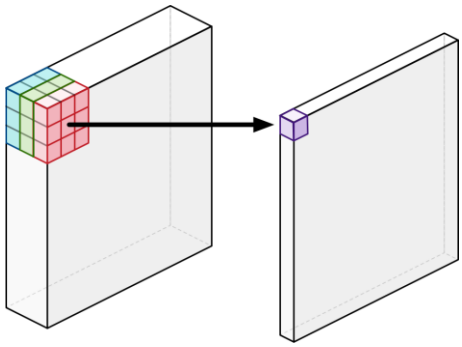
Fully-connected layer adalah lapisan dimana semua neuron aktivitas dari lapisan sebelumnya terhubung semua dengan neuron di lapisan selanjutnya (Lina, Q., 2019). Kemudian pada lapisan terakhir dari arsitektur CNN menggunakan fungsi aktivasi *sigmoid* atau *softmax* untuk menyediakan luaran klasifikasi. Fungsi aktivasi *sigmoid* digunakan untuk *dataset* yang memiliki 2 kelas, sedangkan fungsi aktivasi *softmax* untuk *dataset* yang memiliki 3 kelas atau lebih.

2.5 MobileNetV2

MobileNets merupakan salah satu arsitektur CNN yang dapat digunakan untuk mengatasi kebutuhan akan *computing resource* berlebih. Seperti namanya,

Mobile, para peneliti dari Google membuat arsitektur CNN yang dapat digunakan untuk ponsel. Perbedaan mendasar antara arsitektur MobileNet dan arsitektur CNN pada umumnya adalah penggunaan *layer* konvolusi dengan ketebalan filter yang sesuai dengan ketebalan dari *input image* (Ekoputris, R. O., 2018).

Convolutional layer reguler/biasa menerapkan kernel konvolusi (atau "filter") ke semua *channels* gambar *input*. Ia menggeser kernel ini melintasi gambar dan pada setiap langkah melakukan penjumlahan tertimbang dari piksel *input* yang dicakup oleh kernel di semua *input channels*. Operasi konvolusi menggabungkan nilai semua *input channel*. Jika gambar memiliki 3 *input channels*, maka menjalankan kernel *single convolution* pada gambar ini akan menghasilkan gambar *output* dengan hanya 1 *channel* per piksel. Jadi untuk setiap piksel *input*, tidak peduli berapa banyak *channels* yang dimilikinya, *convolution* menulis piksel *output* baru dengan hanya satu *channel* (Hollemans, M., 2017). Berikut Gambar 6 merupakan ilustrasi *standard convolution*.

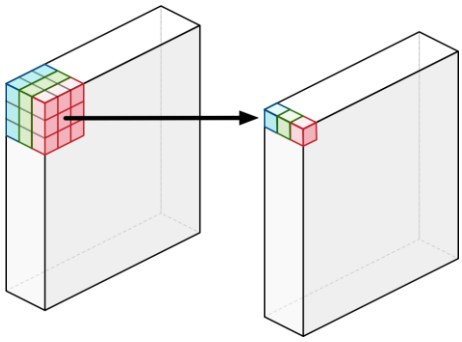


Gambar 6 *Standard convolution*
Sumber: Hollemans, M. (2017)

Arsitektur MobileNetV2 juga menggunakan konvolusi standar ini, tetapi hanya dua kali. Semua *layer* lainnya melakukan *depthwise separable convolution* (DSC) yang merupakan kombinasi dari dua operasi konvolusi yang berbeda, yaitu *depthwise convolution* dan *pointwise convolution*.

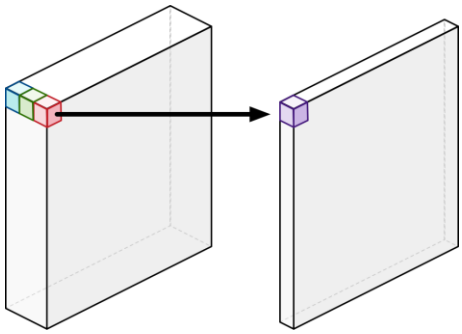
Depthwise convolution ini tidak menggabungkan *input channel* tetapi melakukan konvolusi pada setiap *channel* secara terpisah. Untuk gambar dengan 3 *channels*, *depthwise convolution* menghasilkan gambar *output* yang juga memiliki 3 *channels*. Setiap *channel* mempunyai bobot tersendiri. Tujuan dari *depthwise convolution* adalah untuk menyaring *input channels*. Misalnya deteksi tepi,

pemfilteran warna, dan sebagainya (Hollemans, M., 2017). Berikut Gambar 7 merupakan ilustrasi *depthwise convolution*.



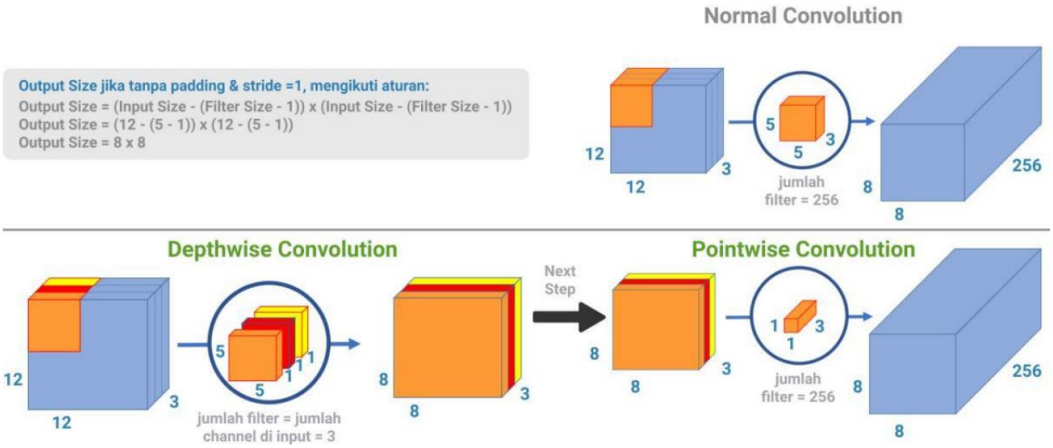
Gambar 7 *Depthwise convolution*
Sumber: Hollemans, M. (2017)

Depthwise convolution diikuti oleh *pointwise convolution*. *Pointwise convolution* sama dengan konvolusi biasa tetapi dengan kernel 1x1. Dengan kata lain, *pointwise convolution* hanya menjumlahkan semua *channels* (sebagai jumlah tertimbang). Tujuan dari *pointwise convolution* adalah untuk menggabungkan *output channels* dari *depthwise convolution* untuk membuat *feature* baru (Hollemans, M., 2017). Berikut Gambar 8 merupakan ilustrasi *pointwise convolution*.



Gambar 8 *Pointwise convolution*
Sumber: Hollemans, M. (2017)

Pada Allaam, M. R. R., & Wibowo, A. T. (2021) memberikan contoh perbandingan jumlah komputasi antara *regular convolution* dibandingkan DSC berdasarkan Gambar 9 berikut.



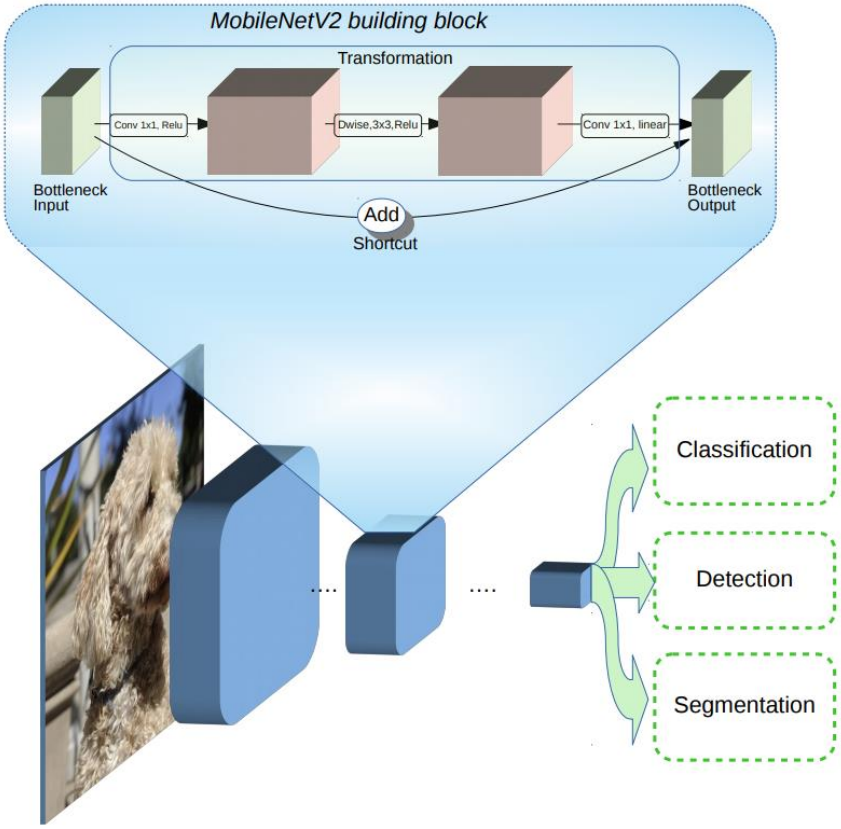
Gambar 9 Perbandingan komputasi konvolusi reguler dan DSC
Sumber: Allaam, M. R. R., & Wibowo, A.T. (2021)

Pada *normal/regular convolution*, terdapat 256 filter 5 x 5 x 3 yang bergeser sebanyak 8 x 8 kali (8 x 8 yaitu jumlah pergeseran filter 5 x 5 dari kiri atas ke kanan bawah terhadap *input image* 12 x 12). Ini berarti 256 x 5 x 5 x 3 x 8 x 8 = 1.228.800 total perkalian yang dilakukan saat proses konvolusi. Sedangkan pada DSC, tahap *depthwise convolution* terdapat 3 filter 5 x 5 x 1 yang bergeser sebanyak 8 x 8 kali. Ini berarti 3 x 5 x 5 x 1 x 8 x 8 = 4.800. Selanjutnya, pada tahap *pointwise convolution* terdapat 256 filter 1 x 1 x 3 yang bergeser sebanyak 8 x 8 kali. Ini berarti 256 x 1 x 1 x 3 x 8 x 8 = 49.152. Maka total perkalian yang dilakukan saat proses konvolusi di DSC ini yaitu 4.800 + 49.152 = 52.952. Dapat dilihat bahwa jumlah 52.952 hanya kurang lebih 22,7% nya saja dari 1.228.800 di *normal convolution* (Allaam, M. R. R., & Wibowo, A. T., 2021).

Konvolusi reguler melakukan pemfilteran dan penggabungan sekaligus, tetapi dengan DSC, kedua operasi ini dilakukan sebagai langkah terpisah. Hasil akhir dari kedua pendekatan ini cukup mirip (keduanya memfilter data dan membuat *feature* baru) namun konvolusi reguler harus melakukan lebih banyak pekerjaan komputasi untuk mencapainya dan perlu mempelajari lebih banyak bobot. Jadi meskipun (kurang lebih) melakukan hal yang sama, DSC akan jauh lebih cepat (Hollemans, M., 2017).

MobileNet merilis versi keduanya pada April 2017 lalu. MobileNetV2 dibangun berdasarkan ide-ide dari MobileNetV1, menggunakan DSC sebagai *building blocks* yang efisien. MobileNetV2 menambahkan dua fitur baru yaitu: 1) *linear bottleneck* antar *layer*, dan 2) *shortcut connections* antar *bottlenecks*

(Ekoputris, R. O., 2018). Struktur dasar dari arsitektur ini ditunjukkan pada Gambar 10 berikut, dimana blok biru menunjukkan blok pembentukkan konvolusi *linear bottleneck.*



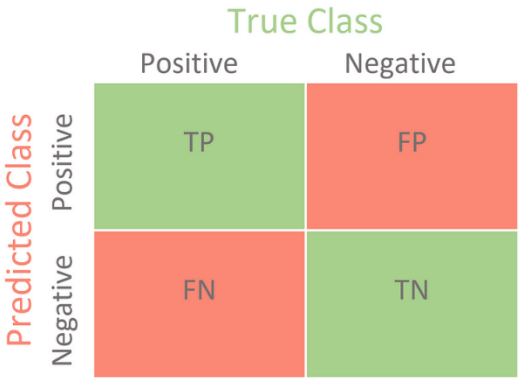
Gambar 10 Gambaran umum arsitektur MobileNetV2
Sumber: Sandler, M. & Howard, A. (2018)

Pada bagian *bottleneck* terdapat *input* dan *output* antara model, sedangkan *layer* bagian dalam meng-enkapsulasi kemampuan model untuk mengubah *input* dari konsep tingkat yang lebih rendah (i.e. piksel) ke deskriptor tingkat yang lebih tinggi (i.e. kategori gambar). Pada akhirnya, seperti halnya *residual connections* pada CNN tradisional, *shortcuts* antar *bottlenecks* memungkinkan *training* lebih cepat dan akurasi yang lebih baik (Ekoputris, R. O., 2018).

2.6 *Confusion Matrix*

Confusion matrix merupakan suatu alat untuk mengukur kinerja dari model klasifikasi yang telah dibangun. *Confusion matrix* menjadi salah satu evaluator untuk mengukur seberapa baik model yang dibangun mampu mengenali kelas-

kelas yang berbeda (Shidiq, A. L. A., dkk., 2022). Berikut Gambar 11 merupakan tabel *confusion matrix*.



Gambar 11 Tabel *confusion matrix*
Sumber: Mohajon, J. (2020)

Keterangan:

a. TP (*True Positive*) merupakan data positif yang diprediksi benar (positif).
b. TN (*True Negative*) merupakan data negatif yang diprediksi benar (negatif).
c. FP (*False Positive*) merupakan data negatif namun diprediksi sebagai data positif.
d. FN (*False Negative*) merupakan data positif namun diprediksi sebagai data negatif.

Berdasarkan hasil *confusion matrix*, maka dapat dihitung indikator evaluasi seperti akurasi. Akurasi merupakan tingkat ketepatan prediksi benar dari keseluruhan data (Shidiq, A. L. A., dkk., 2022). Perhitungan akurasi ditunjukkan pada Persamaan (1) berikut.

$$Akurasi\ (\%) = \frac{TP+TN}{TP+TN+FP+FN} \times 100\% \quad (1)$$

2.7 TensorFlow

TensorFlow (TF) adalah *framework open source* untuk *machine learning* yang dikembangkan Google dan dirilis secara publik pada tahun 2015. TF memudahkan pembuatan model ML bagi pemula maupun ahli. TF dapat dipakai untuk *deep learning*, *computer vision*, pemrosesan bahasa alami (*Natural Language Processing*), serta *reinforcement learning*.

TF disebut sebagai *end-to-end platform* untuk *machine learning* karena TF menyediakan semua *tools* dan *library* yang dibutuhkan untuk proyek *machine*

learning dari tahap *training* model hingga tahap produksi. Beberapa keunggulan TensorFlow sebagai berikut (Setiani, T. D., dkk., 2023).

- Bisa dijalankan di hampir semua *platform*: GPU, CPU, dan TPU (TensorFlow *Processing Units*) yang secara khusus dimanfaatkan untuk mengerjakan matematika tensor.
- Memberikan performa terbaik dengan kemampuan melakukan iterasi dan melatih model secara cepat sehingga mampu menjalankan lebih banyak eksperimen.
- Skalabilitas komputasi yang tinggi pada kumpulan data yang sangat besar.
- Pembuatan model yang mudah dengan beberapa level abstraksi sesuai kebutuhan.
- Menyediakan jalur langsung ke produksi, baik itu pada server, perangkat *mobile* atau web sehingga memudahkan pengguna untuk melakukan *pipeline machine learning* hingga ke level produksi.

2.7.1 TensorFlow Lite

TensorFlow Lite merupakan versi ringan dan efisien dari *framework* TensorFlow yang sering digunakan ML *Developer* untuk mengembangkan dan men-*deploy* model. TensorFlow Lite didesain sedemikian rupa sehingga memungkinkan untuk menjalankan model pada perangkat dengan *resource* yang terbatas, seperti *handphone* dan *embedded system*. Selain itu, TensorFlow Lite mendukung berbagai jenis *machine learning*, seperti pengenalan gambar, pengenalan audio, dan *natural language* (Faizin, A. A., 2023). Melalui laman resmi pada TensorFlow (2022), menyebutkan fitur utama pada TensorFlow Lite sebagai berikut.

- Dioptimalkan untuk *machine learning* pada perangkat, dengan mengatasi 5 kendala utama: latensi (tidak perlu bolak-balik ke server), privasi (tidak ada data pribadi yang keluar dari perangkat), konektivitas (koneksi internet tidak diperlukan), ukuran (model yang dikurangi dan ukuran biner) dan konsumsi daya (inferensi efisien dan kurangnya koneksi jaringan).
- *Multiple platform support*, mencakup perangkat Android dan iOS, *embedded* Linux, dan mikrokontroler.

- Dukungan bahasa yang beragam, termasuk Java, Swift, Objective-C, C++, dan Python.
- Performa tinggi, dengan akselerasi perangkat keras dan optimalisasi model.
- *End-to-end examples*, untuk tugas *machine learning* umum seperti klasifikasi gambar, deteksi objek, estimasi pose, menjawab pertanyaan, klasifikasi teks, dll. di berbagai *platform*.

Secara singkat pada TensorFlow (2023) menyebutkan cara menerapkan model ML di perangkat seluler adalah dengan memilih model baru atau latih ulang model yang sudah ada, kemudian mengonversi model TensorFlow menjadi FlatBuffers (ekstensi file .tflite) terkompresi dengan TensorFlow Lite *Converter*. Selanjutnya lakukan *deploy* atau ambil file .tflite terkompresi dan muat ke perangkat seluler (*mobile interpreter*).

2.7.2 *Library* Keras Tuner

Pada Bag, Sukanya, 2011 menjelaskan bahwa CNN dipengaruhi oleh banyak *hyperparameter* seperti *learning rate*, *batch size*, dan lainnya. Masing-masing *hyperparameter* memiliki peran dalam menggeneralisasikan model untuk mendapatkan ketahanan tertinggi. Pada saat yang sama untuk mendapatkan nilai *hyperparameter* yang tepat diperlukan proses *trial-and-error* dalam jumlah yang besar. Hal ini akan menghabiskan waktu untuk penyempurnaan model *deep learning*. Oleh karena itu, TensorFlow Google menciptakan *framework* yang luar biasa untuk mengatasi kesulitan dalam melakukan penyetelan dan pengoptimalan *hyperparameter*, yaitu menggunakan *library* Keras Tuner.

Keras Tuner adalah *framework* pengoptimalan *hyperparameter* yang mudah digunakan dan dapat diskalakan untuk memecahkan masalah pencarian *hyperparameter* (O'Malley, T., et al., 2019). Keras Tuner memudahkan penentuan ruang pencarian dan memanfaatkan algoritma yang disertakan untuk menemukan nilai *hyperparameter* terbaik. Keras Tuner hadir dengan algoritma Bayesian Optimization, Hyperband, dan Random Search

bawaan, dan juga dirancang agar mudah dikembangkan oleh para peneliti untuk bereksperimen dengan algoritma pencarian baru (O'Malley, T., 2020).

2.8 Google Colaboratory

Google Colaboratory atau disebut juga Colab adalah *tools* yang dikeluarkan oleh Google *Internal Research* yang dibuat untuk membantu para *researcher* dalam mengolah data untuk keperluan belajar maupun bereksperimen pada pengolahan data khususnya bidang *machine learning*. Penggunaan *tools* ini mirip seperti Jupyter Notebook, perbedaannya Colab tidak memerlukan pengaturan atau *setup* terlebih dahulu sebelum digunakan dan berjalan sepenuhnya pada *cloud* dengan memanfaatkan media penyimpanan Google Drive (Digmi, I., 2018).

Colab menyediakan layanan GPU gratis kepada penggunanya sebagai *backend* komputasi dan dapat digunakan selama 12 jam pada suatu waktu. Pengguna tidak perlu mengeluarkan sejumlah uang atau membeli perangkat komputer dengan tambahan GPU jika ingin belajar *machine learning*. Dengan Colab, pengguna dapat membangun aplikasi berbasis *deep learning* menggunakan *library* poluler seperti Keras, TensorFlow, PyTorch dan OpenCV (Digmi, I., 2018).

2.9 Android Studio

Android Studio adalah IDE (*Integrated Development Environment*) berbasis Intellij IDEA yang digunakan khusus untuk membangun aplikasi Android. Bahasa pemrograman yang bisa digunakan untuk mengembangkan aplikasi Android adalah Kotlin dan Java. Android Studio memiliki fitur yang lengkap untuk mengembangkan aplikasi seperti berikut (Ramadhan, G., dkk., 2023).

- *Template*: *template* memulai *project* maupun *Activity* tanpa harus membuatnya dari nol.
- *Intelligent code editor*: *code completion* yang memudahkan untuk menulis kode dengan cepat tanpa harus menuliskan secara lengkap. Selain itu, juga ada *warning* apabila terdapat kesalahan penulisan kode.
- *Design tool*: digunakan untuk mendesain aplikasi beserta melihat *preview* secara langsung sebelum dijalankan.

- *Flexible build system*: Android Studio menggunakan Gradle yang fleksibel untuk menciptakan *build variant* yang berbeda untuk berbagai *device*. Anda juga dapat menganalisa prosesnya secara mendetail.
- *Emulator*: menjalankan aplikasi tanpa harus menggunakan *device* Android. Dilengkapi dengan *Instant Run* untuk melihat perubahan tanpa harus *build project* dari awal.
- *Debugging*: memudahkan untuk mencari tahu masalah.
- *Testing*: menjalankan pengujian untuk memastikan semua kode aman sebelum rilis.
- *Publish*: membuat berkas AAB/APK dan menganalisanya guna dibagikan dan di-*publish* ke PlayStore.
- Integrasi: Terhubung dengan berbagai layanan yang memudahkan untuk mengembangkan aplikasi, seperti Github, Firebase, dan Google Cloud.

2.10 *State of The Art*

Dalam penelitian ini disertakan penelitian terdahulu yang mencakup aspek nasional dan internasional yang berhubungan dengan sistem deteksi cacat kain, implementsi *deep learning* untuk deteksi cacat produk, dan penggunaan TensorFlow Lite. Penelitian tersebut antara lain:

1. Penelitian dengan judul Analisis Metode GLCM dan SVM untuk Mendeteksi Cacat Kain. Diambil dari Jurnal Teknik Komputer, Universitas Komputer Indonesia (Unikom), diteliti oleh Taufik Adi Wicaksono dan John Adler pada tahun 2018 di Bandung, Indonesia. Penelitian ini menggunakan metode statistik dalam mendeteksi cacat kain yaitu metode GLCM dan SVM. GLCM digunakan untuk menganalisis tekstur dan SVM digunakan untuk melakukan prediksi atau mengklasfikasikan pola cacat kain. Jumlah kelas atau jenis cacat kain sebanyak 6 kelas, yaitu lubang, noda, benang putus, *float*, cacat warna dan cacat pola. Hasil yang didapatkan yaitu akurasi sebesar 88,33% pada *training set test* dan 33,33% pada *supplied set test* (Wicaksono, T. A., & Adler, J., 2018).

2. Penelitian dengan judul *Fabric Defect Detection Based on Saliency Histogram Features*. Diambil dari *Computational Intelligence*, diteliti oleh Min Li, Shaohua Wan, Zhongmin Deng dan Yajun Wang tahun 2019 di China. Penelitian ini menggunakan *histogram saliency* dan SVM. *Dataset* yang digunakan sebanyak 700 gambar kain (500 gambar kain bebas cacat dan 200 gambar kain cacat), berasal dari Universitas Hong Kong dan TILDA. Sistem dapat mengklasifikasikan jenis kain dengan 2 kelas yaitu tidak cacat dan cacat (*Broken End, Thin Bar, Thick Bar, Netting Multiple, Knots, Hole, Oil Spot,* dan *Stain*). Hasil yang didapatkan yaitu akurasi sebesar 95,5% dengan waktu komputasi 24,08 ms (Li, M., et al., 2019).
3. Penelitian dengan judul *Fabric Defect Detection Based on Lightweight Neural Network*. Diambil dari Jurnal Zhongyuan University of Technology, diteliti oleh Zhoufeng Liu, Jian Cui, Chunlei Li, Miaomiao Wei, dan Yan Yang tahun 2019, di China. Penelitian ini mengusulkan kerangka kerja berbasis CNN (YOLO-LFD) untuk mendeteksi cacat kain. *Dataset* yang digunakan sebanyak 3000 gambar cacat kain dengan lima kelas cacat (*holes, surface debris, oil stains, longitude* dan *weft breakage*), berasal dari *dataset* COCO. Hasil yang didapatkan yaitu akurasi mencapai 97,2% dengan waktu komputasi 51,0 ms (Liu, Z., et al., 2019).
4. Penelitian dengan judul *CNN-Based Minor Fabric Defects Detection*. Diambil dari *International Journal of Clothing Science and Technology*, diteliti oleh Zhijie Wen, Qikun Zhao dan Lining Tong tahun 2020 di China. Penelitian ini mengusulkan algoritma PETM-CNN untuk deteksi cacat kain *minor*. Algoritma PE (*Patches Extractor*) mengekstraksi tambalan yang mungkin merupakan tambalan yang rusak untuk memproses gambar kain terlebih dahulu. Kemudian dirancang metode TM-CNN (*Triplet Metric* CNN) untuk memprediksi label *patch* dan label akhir *image*. *Dataset* yang digunakan adalah *Minor Fabrics Defects Images* (MFDI) terdiri dari 100 gambar kain normal, dan 100 gambar kain cacat *minor*. Hasil yang didapatkan yaitu akurasi 79,32% (Wen, Z., et al., 2020).

5. Penelitian dengan judul Deteksi dan Klasifikasi Kaleng Menggunakan *Convolutional Neural Network*. Diambil dari jurnal PROZIMA (*Productivity, Optimization and Manufacturing System Engineering*), diteliti oleh Rindi Kusumawardani dan Putu Dana Karningsih tahun 2020 di Surabaya, Indonesia. Penelitian ini mendeteksi cacat pada kemasan kaleng dan klasifikasi jenis cacat pada kaleng secara otomatis untuk mengurangi *human error* pada proses inspeksi manual. *Dataset* yang digunakan sebanyak 600 citra yang terdiri dari 3 kelas (*no defect, minor defect* dan *major defect*). Hasil terbaik yang didapatkan yaitu menggunakan arsitektur ResNet50 dengan nilai akurasi pelatihan sebesar 93,81%, akurasi validasi sebesar 82,46% dan akurasi data tes sebesar 95,56% serta waktu pelatihan 675 menit 16 detik (Kusumawardani, R., & Karningsih, P. D., 2021).
6. Penelitian dengan judul Sistem Deteksi Cacat *Flash* untuk Kontrol Proses *Friction Stir Welding* (FSW) pada Material Aa6061-T651 dengan Metode *Convolutional Neural Network* (CNN). Diambil dari tesis Institut Teknologi Sepuluh Nopember, diteliti oleh Ulya Ganeswara Alamy tahun 2022 di Surabaya, Indonesia. Penelitian ini memberikan kontribusi pada metode *visual inspection* untuk mengontrol hasil proses *Friction Stir Welding* (FSW) dengan mendeteksi cacat *flash* pada material AA6061-T651 menggunakan arsitektur AlexNet dan VGG16. *Dataset* yang digunakan sebanyak 620 citra dari proses FSW diolah menjadi dua kelompok dataset. Hasil yang diperoleh model AlexNet yaitu akurasi deteksi cacat *flash* sebesar 91,03%, sedangkan model VGG16 yaitu akurasi deteksi cacat *flash* sebesar 77,35% (Alamy, U. G., 2022).
7. Penelitian dengan judul *Industrial Laser Welding Defect Detection and Image Defect Recognition Based on Deep Learning Model Developed*. Diambil dari jurnal MDPI, diteliti oleh Honggui Deng, dkk. tahun 2021 di China. Penelitian ini melakukan deteksi cacat produk pengelasan dengan menggunakan 5200 gambar cacat las yang terbagi menjadi 5 kelas (*gas pores, cracks, lack of fusion, lack of penetration* dan *flawless*). Hasil yang didapatkan yaitu akurasi 98,75% (Deng, H., et al., 2021).

8. Penelitian dengan judul Sistem Klasifikasi Penyakit Tanaman Cabai Menggunakan Metode *Deep Learning* dengan *Library TensorFlow Lite*. Diambil dari skripsi Universitas Hasanuddin, diteliti oleh Ahmad Kurniawan Syarif tahun 2021 di Makassar, Indonesia. Penelitian ini membuat sistem klasifikasi penyakit tanaman cabai yaitu Virus Gemini dan Keriting Mozaik dengan menggunakan CNN dengan *library* TensorFlow Lite dan arsitektur EfficientNet, kemudian diintegrasikan pada perangkat *mobile*. Dataset yang digunakan sebanyak 300 citra yang terbagi dalam 3 kelas (sehat, keriting mozaik dan virus gemini). Hasil yang diperoleh yaitu akurasi *training* dan validasi sebesar 96,67% dan hasil testing sebesar 96,67% (Syarif, A. K., 2021).

Berdasarkan penelitian yang telah disebutkan, penelitian Wicaksono, T. A., & Adler, J. (2018) memberikan informasi bahwa hasil akurasi dipengaruhi oleh perbedaan kualitas citra dan adanya nilai interval yang besar antara ciri yang satu dengan yang lainnya. Penelitian Li, M., et al. (2019); Liu, Z., et al. (2019); Wen, Z., et al. (2020) memberikan kontribusi pengetahuan dalam mendeteksi cacat kain dengan metode yang berbeda. Penelitian Kusumawardani, R., & Karningsih, P. D. (2021); Alamy, U. G. (2022); Deng, H., et al. (2021) juga memberikan informasi pengetahuan bahwa metode CNN dapat mendeteksi cacat produk atau mengklasifikasikan citra dengan baik, sehingga menjadi dasar kerangka pikir pada penelitian ini dengan menggunakan metode CNN. Perbedaannya pada penelitian ini akan menggunakan citra kain sebagai objek penelitian. Kemudian penelitian Syarif, A. K. (2021) memberikan kontribusi pengetahuan dalam implementasi model CNN agar dapat diintegrasikan ke dalam perangkat *mobile* (android) yaitu dengan konversi model ke format TensorFlow Lite.