IMPLEMENTASI DAN KLASIFIKASI JENIS-JENIS BATIK MENGGUNAKAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK (CNN) DENGAN MODEL ARSITEKTUR RESNET

SKRIPSI

Digunakan Sebagai Syarat Maju Ujian Diploma IV

Politeknik Negeri Malang

Oleh:

RIZAL WHISNU WIRATAMA **NIM. 1941720227**

PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2023

HALAMAN PENGESAHAN

IMPLEMENTASI DAN KLASIFIKASI JENIS-JENIS BATIK MENGGUNAKAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK (CNN) DENGAN MODEL ARSITEKTUR RESNET

Disusun oleh:

RIZAL WHISNU WIRATAMA **NIM. 1941720227**

Laporan Akhir ini telah diuji pada tanggal 27 Juli 2023

Disetujui oleh:

1. Pembimbing Utama : Dr. Eng. Rosa Andrie Asmara, ST., MT.
   NIP. 198010102005011001 .....
2. Pembimbing Pendamping : Dika Rizky Yunianto, S.Kom., M.Kom
   NIP. 199206062019031017 .....
3. Penguji Utama : Dr. Ulla Delfana Rosiani, ST., MT.
   NIP. 197803272003122002 .....
4. Penguji Pendamping : Milyun Ni'ma Shoumi, S.Kom., M.Kom
   NIP. 196211281988111001 .....

Mengetahui,

Ketua Jurusan
Teknologi Informasi

Dr. Eng. Rosa Andrie Asmara, ST., MT.
NIP. 198010102005011001

Ketua Program Studi
Teknik Informatika

Dr. Ely Setyo Astuti, S.T., M.T.
NIP. 197605152009122001

PERNYATAAN

Dengan ini saya menyatakan bahwa pada Skripsi ini tidak terdapat karya, baik seluruh maupun sebagian, yang sudah pernah diajukan untuk memperoleh gelar akademik di Perguruan Tinggi manapun, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini serta disebutkan dalam daftar sitasi/pustaka.

Malang, 27 Juli 2023

Rizal Whisnu W.

ABSTRAK

Wiratama W., Rizal. "Implementasi dan Klasifikasi Jenis-Jenis Batik Menggunakan Algoritma Convolutional Neural Network Dengan Model Arsitektur ResNet". **Pembimbing: (1) Dr. Eng. Rosa Andrie Asmara, ST., MT., (2) Dika Rizky Yunianto, S.Kom., M.Kom**

Skripsi, Program Studi Teknik Informatika, Jurusan Teknologi Informasi, Politeknik Negeri Malang, 2023.

Penelitian ini bertujuan untuk mengimplementasikan metode deep learning menggunakan Convolutional Neural Network (CNN) dengan model ResNet untuk mengenali jenis-jenis batik. Kesenian batik merupakan salah satu hasil kebudayaan masyarakat Indonesia yang memiliki nilai budaya yang tinggi. Namun, identifikasi motif batik yang beragam seringkali sulit, terutama bagi masyarakat awam. Oleh karena itu, penggunaan kecerdasan buatan dalam bentuk AI dapat memudahkan proses pengenalan pola batik. Pemanfaatan AI dalam bidang pengolahan citra digital sangatlah penting dalam berbagai aplikasi saat ini. Deep learning, sebagai bagian dari AI, memungkinkan komputer untuk belajar mengklasifikasikan objek langsung dari gambar. Metode deep learning ini memanfaatkan CPU, RAM, dan GPU untuk memproses komputasi data besar dengan cepat dan efisien. Convolutional Neural Network (CNN) adalah salah satu metode deep learning yang paling efektif dalam pengenalan citra digital. Arsitektur ResNet, yang merupakan keluarga arsitektur CNN yang dikembangkan oleh Google Research, terbukti memiliki tingkat akurasi yang tinggi dan efisiensi yang lebih baik daripada arsitektur lainnya. Model ResNet memiliki ukuran yang relatif lebih kecil dan waktu inferensi yang lebih cepat, serta mudah disesuaikan untuk berbagai tugas pembelajaran transfer. Dalam konteks penelitian ini, dilakukan implementasi metode deep learning dengan menggunakan CNN dan arsitektur ResNet untuk mengklasifikasikan citra batik. Penelitian ini diharapkan dapat membantu pengenalan jenis-jenis batik, yang pada gilirannya dapat meningkatkan pemahaman dan apresiasi terhadap kebudayaan Indonesia.

Kata Kunci : Pengolahan Citra, Batik, Convolutional Neural Network, ResNet

ABSTRACT

Wiratama W., Rizal. *"Implementation and Classification of Batik Types Using the Convolutional Neural Network Algorithm With the ResNet Architecture Model".* ***Supervisor: Dr. Eng. Rosa Andrie Asmara, ST., MT., Co–Supervisor: Dika Rizky Yunianto, S.Kom., M.Kom***

Thesis, Informatics Management Study Program, Department of Information Technology, State Polytechnic of Malang, 2023.

This study aims to implement the deep learning method using the Convolutional Neural Network (CNN) with the ResNet model to identify types of batik. Batik art is one of the results of the culture of the Indonesian people who have high cultural values. However, identification of various batik motifs is often difficult, especially for ordinary people. Therefore, the use of artificial intelligence in the form of AI can facilitate the batik pattern recognition process. The use of AI in the field of digital image processing is very important in today's various applications. Deep learning, as part of AI, allows computers to learn to classify objects directly from images. This deep learning method utilizes CPU, RAM, and GPU to process large data computations quickly and efficiently. Convolutional Neural Network (CNN) is one of the most effective deep learning methods in digital image recognition. The ResNet architecture, which is a CNN architecture family developed by Google Research, is proven to have a high level of accuracy and better efficiency than other architectures. The Resnet model has a relatively smaller size and faster inference time, and is easily adapted for a variety of transfer learning tasks. In the context of this research, the implementation of the deep learning method uses CNN and the Resnet architecture to classify batik images. This research is expected to help identify the types of batik, which in turn can increase understanding and appreciation of Indonesian culture.

Keywords: *Image Processing, Batik, Convolutional Neural Network, Resnet*

KATA PENGANTAR

Puji Syukur kami panjatkan kehadirat Allah SWT/Tuhan YME atas segala rahmat dan hidayah-Nya penulis dapat menyelesaikan skripsi dengan judul "Implementasi Dan Klasifikasi Jenis-Jenis Batik Menggunakan Algortima Convolutional Neural Network (CNN) Dengan Model Arsitektur ResNet". Skripsi ini penulis susun sebagai persyaratan untuk menyelesaikan studi program Diploma IV Program Studi Teknik Informatika, Jurusan Teknologi Informasi, Politeknik Negeri Malang.

Kami menyadari bahwasannya dengan tanpa adanya dukungan dan kerja sama dari berbagai pihak, kegiatan laporan akhir ini tidak akan dapat berjalan baik. Untuk itu, kami ingin menyampaikan rasa terima kasih kepada:

1. Bapak Dr. Eng. Rosa Andrie Asmara, ST., MT., selaku Ketua Jurusan Teknologi Informasi serta menjadi Dosen Pembimbing 1.
2. Ibu Dr. Ely Setyo Astuti, ST., MT., selaku Sekretaris Jurusan Teknologi Informasi
3. Ibu Mungki Astiningrum, ST., M.Kom., selaku Ketua Program Studi DIV Teknik Informatika
4. Bapak Dika Rizky Yunianto S.Kom., M.Kom, selaku Dosen Pembimbing 2.
5. Kedua orang tua penulis, Bapak Sukirno dan Ibu Yayuk Asriningdyah, yang selalu memberikan kasih sayang, doa, nasehat, serta atas kesabarannya yang luar biasa dalam setiap langkah penulis, yang merupakan anugerah terbesar dalam hidup. Sehingga kelak penulis akan menjadi anak yang dibanggakan.
6. Dan seluruh pihak yang telah membantu dan mendukung lancarnya pembuatan Laporan Akhir dari awal hingga akhir yang tidak dapat kami sebutkan satu persatu.

Penulis menyadari bahwa dalam penyusunan laporan akhir ini, masih banyak terdapat kekurangan dan kelemahan yang dimiliki penulis baik itu sistematika penulisan maupun penggunaan bahasa. Untuk itu penulis mengharapkan saran dan kritik dari berbagai pihak yang bersifat membangun demi penyempurnaan laporan ini. Semoga laporan ini berguna bagi pembaca secara umum dan penulis secara khusus. Akhir kata, penulis ucapkan banyak terima kasih.

Malang, 27 Juli 2023

Rizal Whisnu W.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I. PENDAHULUAN

1.1. Latar Belakang

Indonesia merupakan negara dengan jumlah populasi penduduk lebih dari 255 juta jiwa pada tahun 2015 dan menempati posisi nomor lima jumlah populasi penduduk terbesar di dunia. Indonesia juga merupakan negara kepulauan terbesar di dunia yang memiliki 13.466 pulau. Wilayah Indonesia terbentang sepanjang 3.977 mil di antara Samudera Hindia dan Samudera Pasifik dengan luas daratan 1.922.570 km2 dan luas perairan 3.257.483 km2 sehingga Indonesia kaya akan keanekaragaman suku, ras, gama, budaya, adat dan bahasa daerah. (Statistik Indonesia (2002). Dari data tersebut telah membuktikan bahwa Indonesia memiliki keanekaragaman budaya sehingga menjadikan suatu kebanggaan dan memiliki keunikan tersendiri untuk dilestarikan (Triyani et al., 2019).

Salah satu hasil kebudayaan masyarakat Indonesia adalah Batik. Indonesia sebagai negara dengan keberagaman budaya memiliki motif batik yang khas disetiap daerah (Arsa & Susila, 2019). Saat ini ada ratusan motif kain batik yang tersebar di Indonesia yang memiliki nama dan makna tersediri dengan ciri khas motif yang membedakannya. Banyaknya pola batik di Indonesia mengakibatkan sulitnya untuk mengindentifikasi motif batik, khususnya pada masyarakat awam. Tapi banyak juga masyarakat yang sudah bisa membedakan berbagai jenis motif batik. Dan masalah utamanya ada pada tingkat akurasi dalam mengklasifikasi gambar jenis-jenis motif batik yang masih banyak perbedaan pada hasil akurasi jenis-jenis motif batik tersebut. Tujuan dari dilakukan nya penelitian ini adalah untuk mengetahui atau membuktikan dengan metode yang akan digunakan dalam penelitian ini dapat mendapatkan hasil akurasi yang lebih baik dari penelitian-penelitian sebelumnya. Hal ini menjadi inspirasi para peneliti dibidang ilmu komputer untuk melakukan penelitian terkait klasifikasi batik yang relatif susah untuk dibedakan menggunakan mata manusia (Samsi & Soedewi, 2007). Diperlukan pengalaman dan pengetahuan khusus dibidang khazanah perbatikan nusantara untuk mengenali asal daerah dan jenis dari suatu motif batik daerah tertentu. Kemajuan ilmu komputer saat ini khususnya di bidang computer vision, klasifikasi citra dapat dilakukan menggunakan metode-metode kecerdasan buatan.

Banyaknya motif di Indonesia menyulitkan untuk identifikasi jenisnya. Penelitian yang dilakukan oleh (Mawan, 2020) bertujuan untuk mengetahui motif dengan bantuan komputasi dapat membantu dalam pelestarian batik. Convolutional Neural Network (CNN) adalah salah satu metode machine learning dari pengembangan Multi Layer Perceptron (MLP) yang didesain untuk mengolah data dua dimensi. CNN termasuk dalam jenis Deep Neural Network karena dalamnya tingkat jaringan dan banyak diimplementasikan dalam data citra. Eksperimen menggunakan Dataset 120 potongan foto Batik (3 kelas) menunjukkan bahwa model yang menggunakan CNN mencapai rata-rata akurasi 65% sedangkan model CNN dikombinasi dengan Grayscale mencapai rata-rata akurasi 70%. Meskipun demikian dengan penambahan Grayscale akurasi bertambah 5%. Dari dataset yang berjumlah 120, data terbagi menjadi dua yaitu Data training 90 dan data uji 30.

Pada penelitian-penelitian sebelumnya yang telah dilakukan oleh (Tumewu et al., 2020), bertujuan untuk mengklasifikasikan motif batik umumnya menggunakan motif-motif geometri saja dengan menggunakan Convolutional Neural Network (CNN) VGG dan penerapan augmentasi sebatas pada skala dan rotasi. Untuk itu pada penelitian tersebut akan dilakukan augmentasi yang lebih beragam pada model CNN Residual Network (Resnet) dengan objek yang dijadikan penelitian adalah motif geometri dan non-geometri. Kemudian pada penelitian didapatkan hasil pengujian menunjukkan CNN dengan penggunaan data augmentation pada training dataset memberikan akurasi hingga 84,52% pada Resnet-18 dan 81,90% pada Resnet-50. Adapun pada penggunaan dataset dengan augmentasi rotation memberikan peningkatan akurasi sebesar 4,06%, augmentasi random erase memberikan peningkatan sebesar 9,38%, augmentasi scale sebesar 6,52%, dan pada augmentasi flip sebesar 8,58%.

Residual Network atau yang biasa disebut sebagai *ResNet* merupakan salah satu arsitektur dari CNN yang diusulkan oleh He, dkk. pada tahun 2015. Arsitektur ini dibagun untuk mengatasi permasalahan pada pelatihan DL, karena pelatihan DL pada umumnya memakan cukup banyak waktu dan terbatas pada jumlah lapisan tertentu. Solusi permasalahan yang diusulkan oleh *ResNet* adalah dengan menerapkan *skip connection* atau *shortcut*. Kelebihan model *ResNet* dibandingkan dengan model arsitektur CNN yang lain adalah kinerja dari model ini tidak menurun

walaupun arsitekturnya semakin dalam. Jadi dilakukan nya penelitian ini agar dapat mengklasifikasikan citra batik dengan hasil yang ditargetkan bisa cukup maksimal.

Sehingga dalam penelitian ini berfokus terhadap bagaimana menaikkan akurasi dalam klasifikasi citra pada pola batik menggunakan arsitektur *Convolutional Neural Network* (CNN) yaitu *Residual Network* atau biasa disebut ResNet (Hasan Mahmud & al Faraby, 2019). Beberapa jenis dari arsitektur ResNet di antaranya adalah ResNet50, ResNet101, dan ResNet152. ResNet menggunakan lebih banyak data dengan teknik augmentasi data. ResNet dipilih sebagai arsitektur CNN karena menurutnya ResNet lebih mudah untuk dioptimalkan dan dapat memperoleh akurasi tinggi. Penelitian klasifikasi motif batik ini menggunakan metode CNN dengan arsitektur Resnet untuk mengenali pola pada motif batik. Selain itu, pada dataset akan diterapkan augmentasi Scale, Random Erase, Rotation, dan Flip. Augmentasi pada dataset diterapkan guna memperbesar jumlah dan meningkatkan variasi pada dataset batik.

Berdasarkan latar belakang di atas, penelitian ini menerapkan implementasi dari metode *deep learning* menggunakan *Convolutional Neural Network* (CNN) dengan arsitektur *ResNet* untuk membantu mengenali jenis-jenis batik. Penelitian ini berfokus terhadap bagaimana mengklasifikasikan citra batik.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, maka rumusan masalah yang dapat diambil adalah sebagai berikut:

1. Bagaimana mengimplementasikan metode *Deep Learning* menggunakan *Convolutional Neural Network* (CNN) untuk mengklasifikasikan citra batik berdasarkan jenis-jenis batik?
2. Bagaimana tingkat akurasi yang didapatkan dari hasil klasifikasi menggunakan *Convolutional Neural Network* (CNN) dengan arsitektur ResNet?

1.3 Batasan Masalah

Adapun batasan masalah dalam skripsi yang berjudul "Implementasi dan Klasifikasi Jenis-Jenis Batik Menggunakan Algoritma *Convolutional Neural Network* (CNN)" adalah sebagai berikut:

1. Penelitian ini ditujukan untuk mengukur tingkat akurasi menggunakan *Convolutional Neural Network* (CNN) dalam klasifikasi citra batik berdasarkan jenis-jenis batik.
2. Data yang digunakan adalah *dataset* Batik Indonesia yang diambil dari laman www.kaggle.com
3. Menggunakan bahasa pemrograman python untuk memproses dan membangun model *Convolutional Neural Network* (CNN)
4. Penelitian ini menggunakan metode ResNet

1.4 Tujuan Penelitian

Tujuan dari dilakukannya skripsi dengan judul "Implementasi dan Klasifikasi Jenis-Jenis Batik Menggunakan Algoritma *Convolutional Neural Network* (CNN)" adalah sebagai berikut:

1. Mengetahui implementasi metode *Deep Learning* menggunakan *Convolutional Neural Network* (CNN) untuk mengklasifikasikan citra batik berdasarkan jenis-jenis batik
2. Mengetahui tingkat akurasi yang didapatkan dari hasil klasifikasi menggunakan *Convolutional Neural Network* (CNN) dengan arsitektur ResNet

1.5 Manfaat

Manfaat dari dilakukannya skripsi dengan judul "Implementasi dan Klasifikasi Jenis-Jenis Batik Menggunakan Algoritma *Convolutional Neural Network* (CNN)" adalah sebagai berikut:

1. Memberikan pengetahuan mengenai implementasi *deep learning* menggunakan *Convolutional Neural Network* (CNN) untuk klasifikasi citra jenis-jenis batik
2. Mengetahui tingkat akurasi dari implementasi *Convolutional Neural Network* (CNN)
3. Mengklasifikasikan batik berdasarkan jenis-jenis batik
4. Membantu masyarakat dalam mengenali jenis-jenis batik

BAB II. LANDASAN TEORI

2.1 Studi Literatur

Berdasarkan penelitian yang akan dilakukan, acuan dari beberapa penelitian terdahulu menjadi sangat penting dalam melakukan sebuah penelitian dengan tujuan untuk mengetahui hubungan antara penelitian yang akan dilakukan dengan penelitian terdahulu, sehingga dengan menambahkan acuan tersebut dapat menambah referensi, perbedaan, serta keunikan dalam penelitian yang akan dilakukan.

Banyak pengembangan sistem yang menggunakan teknologi *Computer Vision*, seperti: *face detection, image recognition* maupun pengenalan dalam pola tertentu. Pengembangan sistem ini menjadi sebuah fungsionalitas yang akan mempermudah pekerjaan dalam berbagai bidang. Pengembangan dari *deep learning* ini sangat tepat dan efektif dalam menyelesaikan permasalahan tersebut. Hal ini tidak lepas dengan adanya riset atau penelitian dalam bidang tersebut. Penelitian terdahulu mengenai *deep learning* dengan menggunakan *Convolutional Neural Network* (CNN) dilakukan oleh para *reasearch* pada berbagai macam *object*.

Adapun penelitian yang dilakukan oleh Rizky Mawan, Penelitian ini bertujuan mengetahui motif dengan bantuan komputasi dapat membantu dalam pelestarian batik. Convolutional Neural Network (CNN) adalah salah satu metode machine learning dari pengembangan Multi Layer Perceptron (MLP) yang didesain untuk mengolah data dua dimensi. CNN termasuk dalam jenis Deep Neural Network karena dalamnya tingkat jaringan dan banyak diimplementasikan dalam data citra. Eksperimen menggunakan Dataset 120 potongan foto Batik (3 kelas), dari dataset yang berjumlah 120, data terbagi menjadi dua yaitu Data training 90 dan data uji 30. Batik di Indonesia memiliki motif yang berbeda-beda, hal tersebut yang bisa digunakan untuk mengklasifikasikan masing-masing kelas batik. Akurasi yang didapatkan menggunakan metode CNN dan CNN kombinasi dengan Grayscale memiliki tingkat akurasi yang berbeda. Hasil menunjukkan bahwa model yang menggunakan CNN mencapai rata-rata akurasi 65% sedangkan model CNN dikombinasi dengan Grayscale mencapai rata-rata akurasi 70%. Jadi Nilai akurasi

terbaik didapatkan dengan metode CNN kombinasi grayscale dengan tingkat akurasi sebesar 70%. Meskipun demikian dengan penambahan Grayscale akurasi bertambah 5%. Nilai hasil akurasi dari penelitian ini tentu masih memungkinkan ditingkatkan dengan mencari metode klasifikasi terbaik untuk motif batik. Penelitian ke depan dengan metode lain, perubahan tekstur gambar ke bentuk lain dan penambahan jumlah data set bisa dilakukan untuk mendapatkan hasil yang lebih baik.

Adapun penelitan yang dilakukan oleh Ardian Yusuf Wicaksono, (Wicaksono et al., 2017) mengenai Modifikasi Arsitektur Convolutional Neural Network untuk klasifikasi motif gambar batik. Penelitian yang dilakukan oleh Ardian Yusuf Wicaksono, dkk menggunakan metode CNN dengan mengembangkan pada arsitektur dari modelnya dengan mengkombinasi GoogleNet dan Residual Networks yang dinamai IncRes. Penelitian ini menggunakan 11 class dari tipe motif batik dengan jumlah data gambar 7112 yang dibagi kedalam 6401 digunakan untuk data latih (train) dan 711 digunakan untuk data uji (test). Dari hasil penelitian ini memperoleh accuracy sebesar 70.84 % dengan waktu 733 ms (milisecond).

Adapun penelitian yang dilakukan oleh Hendry fonda, dkk (Nugroho et al., 2020), studi ini membahas tentang klasifikasi batik riau yang mempunyai keunikan yaitu dari segi motif yang cenderung menggunakan bunga dan pewarnaan yang lembut. Penelitian ini memanfaatkan 168 gambar (*image*). Klasifikasi batik Riau dengan menggunakan *Tensor flow* dengan proses *training* data terilihat adanya klasifikasi gambar dengan jumlah data 168 gambar dengan 68 gambar berupa batik riau dan 100 gambar merupakan gambar yang bukan batik riau. Untuk proses *iterasi* dengan dengan menggunakan 30 *epoch/iterasi training* data. Dari hasil *iterasi* maka didapat nilai akurasi dari data tersebut adalah 65% dengan nilai kerugian (*loss*) sebesar 2.5% dan 2,1%. Hasil Klasifikasi menggunakan CNN menghasilkan batik riau dan bukan batik riau dengan akurasi 65%. Akurasi 65% disebabkan pada dasarnya banyak motif yang sama antara batik riau dengan batik lainnya dengan perbedaan terletak pada warna cerap pada batik riau.

Adapun penelitian yang dilakukan oleh Lutfi Hakim, dkk (Hakim et al., 2020), Sistem klasifikasi motif batik Banyuwangi merupakan sebuah sistem yang

dibangun dengan menggunakan *library Pytorch* dengan Bahasa pemrograman *Python*. Sistem ini dapat mengenali 7 macam motif batik Banyuwangi yaitu diantaranya Gajah Oling, Gedegan, Kopi Pecah, Moto Pitik, Beras Kutah, Paras Gempal dan Sisikan. Sistem ini menggunakan metode *Convolutional Neural Network* (CNN) dan untuk evaluasi digunakan metode *confusion matrix* untuk mengukur nilai akurasi. Penelitian menggunakan model CNN dengan arsitektur yang diberi nama *MyCustomModel*. Berdasarkan hasil penelitian yang didapatkan dan diuraikan pada bagian sebelumnya didapatkan bahwa hasil pengambilan dataset citra untuk penelitian sebanyak 1024 dengan 7 macam motif batik Banyuwangi yaitu, Gajah oling, Beras kutah, Gedegan, Sisikan, Paras gempal, Moto pitik, dan Kopi pecah. Untuk setiap motif batik mengumpulkan data sebanyak 120 citra. Selain itu, proses *Training* model menggunakn metode CNN dengan menggunakan data *train* sebesar 80% dan pada proses *predict* memakai data *test* sebesar 20%. Proses testing dilakukan menggunakan CPU pada *local device* yang dimiliki oleh peneliti menghasilkan waktu rata rata yang dibutuhkan oleh model *MyCustomModel* untuk memprediksi 1 citra sebesar 0,012 detik. Nilai performansi model dari proses evaluasi menggunakan metode *Confusion matrix* terhadap model *architecture* CNN yang diusulkan mendapatkan nilai akurasi sebesar 63%.

Adapun penelitian yang dilakukan oleh Samuel Febrian Tumewu, dkk (Tumewu et al., 2020), Penelitian klasifikasi motif batik ini menggunakan metode CNN dengan arsitektur Resnet untuk mengenali pola pada motif batik. Motif batik yang menjadi objek penelitian ini adalah motif kategori geometri (ceplok, kawung, lereng, nitik, dan parang) dan kategori nongeometri (semen dan lunglungan). Selain itu, pada dataset akan diterapkan *augmentasi Scale, Random Erase, Rotation*, dan *Flip*. Augmentasi pada dataset diterapkan guna memperbesar jumlah dan meningkatkan variasi pada dataset batik. Hasil pengujian menunjukkan CNN dengan penggunaan data *augmentation* pada *training* dataset memberikan akurasi hingga 84,52% pada Resnet-18 dan 81,90% pada Resnet-50. Adapun pada penggunaan dataset dengan augmentasi rotation memberikan peningkatan akurasi sebesar 4,06%, *augmentasi random erase* memberikan peningkatan sebesar 9,38%, *augmentasi scale* sebesar 6,52%, dan pada *augmentasi flip* sebesar 8,58%.

Adapun penelitian yang dilakukan Amin Padmo Azam Masa, dkk (Azam et al., 2021), Penelitian terkait CNN menggunakan Residual Networks (Resnet) sebagai arsitektur utama untuk mengidentifikasi motif batik. Penelitian identifikasi ini menerapkan percobaan pada dataset berupa *scale, random erase, rotasi, dan flip augmentation*. Hasil penelitian ini menunjukkan penggunaan CNN Resnet dengan *augmentasi* data pada dataset menghasilkan akurasi 84,52% dalam Resnet-18 dan menghasilkan akurasi 81,90% dalam Resnet-50. Klasifikasi citra menggunakan arsitektur CNN dalam klasifikasi multi-label pada batik juga telah dilakukan dengan pengenalan objek batik sebanyak 15 motif batik. Hasil dari penelitian ini mencapai 91,41% dengan penggunaan *epoch* sebanyak 100. Penelitian selanjutnya membandingkan pengenalan citra batik menggunakan arsitektur CNN dengan arsitektur CNN yang dikombinasikan *Grayscale* pada citra. Hasil yang diperoleh pada pengenalan citra menggunakan arsitektur CNN mencapai akurasi 70%, dan hasil pada pengenalan citra menggunakan arsitektur CNN dikombinasikan *Grayscale* pada citra mendapatkan pertambahan nilai akurasi sebanyak 5%. Dataset yang digunakan pada penelitian ini adalah sebanyak 120 citra dengan rincian 90 data training dan 30 data uji. Sementara itu, penelitian perbandingan lainnya pada penggunaan arsitektur CNN untuk mengenali motif batik dan model VGG16. Penelitian ini menggunakan dataset sebanyak 300 data yang diambil dari 50 jenis batik. Hasil penelitian menggunakan arsitektur CNN memperoleh nilai akurasi 98% dan membutuhkan waktu yang lebih cepat dibandingkan model VGG16.

Tabel 1 Perbandingan Studi Literatur

No	Penulis	Judul	Arsitektur	Akurasi
1.	(Mawan, Rizki. 2020)	Klasifikasi Motif Batik Menggunakan *Convolutional Neural Network*	*Convolutional Neural Network* (CNN)	65%
			Convolutional Neural Network (CNN) dengan GreyScale	70%
2.	(Wicaksono et al., 2017)	Modifikasi Arsitektur Convolutional Neural Network untuk Klasifikasi Motif Gambar Batik	*Convolutional Neural Network* (CNN) dengan kombinasi GoogleNet dan Residual	70,84%

			Networks yang dinamai IncRes	
3.	(Fonda, H. 2020)	Klasifikasi Batik Riau dengan Menggunakan *Convolutional Neural Network* (CNN)	*Convolutional Neural Network* (CNN)	65%
4.	(Hakim et al., 2023)	Klasifikasi Citra Motif Batik Banyuwangi Menggunakan Convolutional Neural Network	*Convolutional Neural Network* (CNN)	63%
5.	(Tumewu et al., n.d.)	Klasifikasi Motif Batik menggunakan metode Deep Convolutional Neural Network dengan Data Augmentation	*Convolutional Neural Network* (CNN)	84,52%
6.	(Azam et al., 2021)	Klasifikasi Motif Citra Batik Menggunakan Convolutional Neural Network Berdasarkan K-means Clustering	*Convolutional Neural Network* (CNN) dengan asrsitektur ResNet dan berdasarkan K-means Clustering	98%

Berdasarkan analisa penelitian terdahulu, dapat diambil kesimpulan bahwa pada penelitian yang dilakukan oleh Mawan Rizki, dkk menggunakan arsitektur Convolutional Neural Network (CNN) dengan GreyScale serta pada penelitian yang dilakukan oleh Wicaksono, dkk dengan menerapkan Convolutional Neural Network (CNN) dengan kombinasi GoogleNet dan Residual Networks yang dinamai IncRes serta menerapkan data augmentasi yang dilakukan oleh Tumewu, dkk. Dan menerapkan asrsitektur ResNet dan berdasarkan K-means Clustering yang dilakukan oleh Azam, dkk. Diharapkan penelitian pada Implementasi Image Classification Pada Jenis-Jenis Batik Menggunakan Arsitektur Convolutional Neural Network akan mendapatkan nilai akurasi yang baik.

2.2 Dasar Teori

2.2.1 Batik

Batik adalah salah satu budaya yang terdapat di berbagai daerah di Indonesia dan harus dilestarikan. Secara etimologi kata batik berasal dari bahasa Jawa, yaitu ”tik” yang berarti titik/ matik (kata kerja, membuat titik) yang kemudian berkembang menjadi istilah ”batik”. Menurut KBBI kata batik kain bergambar yang pembuatannya secara khusus dengan menuliskan atau menerakan malam pada kain itu, kemudian pengolahannya diproses dengan cara tertentu. Menurut Hamzuri dalam (Prasetyo, 2016) batik merupakan suatu cara untuk memberi hiasan pada kain dengan cara menutupi bagian-bagian tertentu dengan menggunakan perintang. Perintang yang sering digunakan yaitu lilin atau malam. Menurut (Sari, Wulandari, & Maya, 2019) kain bermotif batik baru bisa disebut batik jika motif dihasilkan dari proses pembatikan menggunakan malam batik. Sedangkan menurut (Suliyanto, Novandari, & Setyawati, 2015) Batik merupakan teknik menghias kain dengan menggunakan lilin dalam proses pencelupan warna, dimana semua proses tersebut menggunakan tangan. Adapun peralatan pembuatan batik tulis menurut (Sari et al., 2019) canting, gawangan, kompor, wajan, bak celup, panci atau tong, bandul, taplak, saringan malam, dhingklik (tempat duduk), dan pisau.

Menurut (Prasetyo, 2016) adapun jenis-jenis batik yang ada menurut teknik pembuatannya yaitu : 1) Batik Tulis, Yaitu batik yang kain yang dihias dengan tekstur dan corak batik menggunakan tangan. Pembuatan batik tulis ini bisa memakan waktu ±2-3 bulan 2) Batik Cap, Yaitu batik yang dihias dengan tekstur dan corak batik yang dibentuk dengan cap (biasanya terbuat dari tembaga). Proses pembuatan batik jenis ini membutuhkan waktu ±2-3 hari. 3) Batik Lukis Yaitu jenis batik yang proses pembuatan dengan cara langsung melukis pada kain putih. Motif batik sangat berguna untuk memperindah batik itu sendiri. Menurut (Prasetyo, 2016) motif batik adalah corak atau kerangka gambar yang terdapat pada kain batik sehingga menjadi keseluruhan

batik. Motif batik adalah kerangka gambar pada batik berupa perpaduan antara garis, bentuk dan isen menjadi satu kesatuan yang mewujudkan batik secara keseluruhan.

Gambar 1Contoh Gambar Batik

2.2.2 Motif Batik

Keanekaragaman motif batik dari seluruh Indonesia membuat beberapa orang kesulitan untuk mengenalinya. Untuk memudahkan pengenalan, beberapa seniman batik mengelompokkan motif-motif tersebut berdasarkan bentuk geometris setiap motif, yaitu: kelompok dengan ragam hias geometris dan ragam hias non geometris. Batik dengan ragam hias geometris adalah batik dengan dasar berbentuk bangun geometri seperti persegi, persegi panjang, lingkaran, segitiga, dan lainnya. Contoh dari batik geometris yaitu bentuk motif kawung, parang, nitik, ceplok, dan lain sebagainya. Sedangkan, batik dengan ragam hias nongeometris adalah batik dengan unsur dasar bukan bangun geometris. Ragam hias ini cenderung fleksibel dan lebih menceritakan keadaan alam atau masyarakat sekitar dengan bentuk bunga, daun, hewan, dan lainnya. Contoh dari batik nongeometris yaitu batik motif mega mendung (Samsi & Soedewi, 2007).

2.2.3 Klasifikasi Citra

Salah satu permasalahan dalam bidang pengenalan pola adalah klasifikasi citra ke dalam kelas tertentu. Batik dapat diklasifikasikan

berdasarkan bentuk motifnya yaitu motif geometri, motif non geometri dan motif khusus. Motif Citra batik yang sangat beragam menyulitkan dalam pengenalan setiap pola citra batik. Klasifikasi data diperlukan untuk mengidentifikasi karakteristik obyek yang terkandung dalam basis data dan dikategorikan ke dalam kelompok yang berbeda. Tujuan klasifikasi batik adalah membagi citra batik ke dalam kelas-kelas motif sesuai dengan pola motifnya sehingga mudah untuk dikenali sesuai dengan cirinya.

Klasifikasi batik dilakukan melalui serangkaian tahapan mulai dari *preprocessing* hingga proses pengenalan melalui algoritma klasifikasi. Tahapan pengolahan citra yang digunakan untuk mengetahui karakteristik suatu citra yaitu *preprocessing*, segmentasi, ekstraksi, klasifikasi, dan lain-lain. *Preprocessing* yaitu menghilangkan noise, meningkatkan kualitas citra, melakukan perbaikan citra serta menentukan bagian citra yang akan di observasi. Segmentasi yaitu melakukan partisi citra menjadi bagian - bagian pokok yang mengandung informasi penting. Segmentasi dilakukan dengan memisahkan objek dan background dengan hasil yang diperoleh berupa citra biner. Ekstraksi merupakan bahan pembeda antar objek – objek pada tahap pengambilan ciri pada objek sedangkan klasifikasi merupakan algoritma untuk mengklasifikasikan objek berdasarkan karakteristik dari ciri yang diberikan. Hasil yang akan diperoleh dari algoritma ini yaitu objek akan dikelompokan ke dalam kelas yang sesuai dengan ciri dari data tersebut.

Ekstraksi ciri merupakan salah satu proses awal yang penting dalam melakukan klasifikasi citra dalam pengenalan pola. Citra batik yang terklasifikasi dengan baik akan memberikan informasi yang citra batik yang dapat digunakan untuk pelestarian motif batik. Hasil yang diperoleh dalam penelitian ini adalah klasifikasi citra batik berdasarkan tekstur *gray level co occurrence metrices* motif citra batik dengan metode jaringan syaraf tiruan.

2.2.4 Citra Digital

Citra digital adalah gambar dua dimensi yang dihasilkan dari analog dua dimensi yang kontinu menjadi gambar melalui proses sampling. Gambar analog dibagi menjadi N baris dan M kolom sehingga menjadi gambar diskrit. Citra digital merupakan citra yang dapat diolah komputer. Yang disimpan dalam komputer hanyalah angka-angka yang menunjukkan besar intensitas pada masing-masing piksel. Karena berbentuk data numerik, maka citra digital dapat diolah dengan komputer.

Citra yang dimaksud disini adalah gambar diam (foto) maupun gambar bergerak (yang berasal dari webcam). Sedangkan digital disini mempunyai maksud bahwa pengolahan citra/gambar dilakukan secara digital menggunakan komputer. Secara matematis, citra merupakan fungsi kontinyu (continue) dengan intensitas cahaya pada bidang dua dimensi. Agar dapat diolah dengan komputer digital, maka suatu citra harus dipresentasikan secara numerik dengan nilai-nilai diskrit. Repersentasi dari fungsi kontinyu menjadi nilai-nilai diskrit disebut digitalisasi citra. Sebuah citra digital dapat diwakili oleh sebuah matriks dua dimensi f(x,y) yang terdiri dari M kolom dan N baris, dimana perpotongan antara kolom dan baris disebut piksel (pixel = picture element) atau elemen terkecil dari sebuah citra (Kusumanto & Tompunu, 2011).

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,M-1) \\ f(1,0) & f(1,1) & \dots & f(1,M-1) \\ f(N-1,0) & f(N-1,1) & \dots & f(N-1,M-1) \end{bmatrix} \quad (1.1$$

Suatu citra ∫(x,y) dalam fungsi matematis dapat dituliskan sebagai berikut:

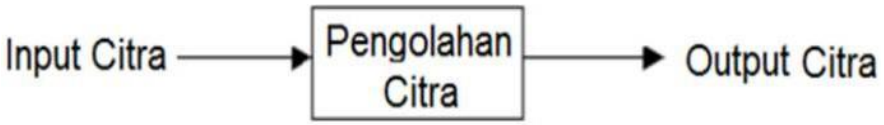
$0 \le x \le M-1$

$0 \le y \le N-1$

2.2.5 Pengolahan Citra

Pengolahan citra digital adalah ilmu yang mempelajari hal-hal berkaitan dengan perbaikan kualitas terhadap suatu gambar

(meningkatkan kontras, perubahan warna, restorasi citra), transformasi gambar (translasi, rotasi transformasi, skala, geometrik), melakukan pemilihan citra ciri (feature images) yang optimal untuk tujuan analisis, melakukan penyimpanan data yang sebelumnya dilakukan reduksi dan kompresi, transmisi data, dan waktu proses data. Pengolahan Citra Digital ini menggunakan bahasa pemrograman *Phyton* yang fungsinya melihat hasil perubahan citra. Adapun diagram sederhana dari proses pengolahan citra dapat dilihat pada gambar di bawah ini (Pengolahan… et al., 2019).



Gambar 2 Tahapan Pengolahan Citra

2.2.6 Machine Learning

Machine learning dapat didefinisikan sebagai aplikasi komputer dan algoritma matematika yang diadopsi dengan cara pembelajaran yang berasal dari data dan menghasilkan prediksi di masa yang akan datang (Goldberg & Holland, 1988). Adapun proses pembelajaran yang dimaksud adalah suatu usaha dalam memperoleh kecerdasan yang melalui dua tahap antara lain latihan (training) dan pengujian (testing) (Huang, Zhu, & Siew, 2006). Bidang *machine learning* berkaitan dengan pertanyaan tentang bagaimana membangun program komputer agar meningkat secara otomatis dengan berdasar dari pengalaman (Mitchell, 1997).

Penelitian terkini mengungkapkan bahwa *machine learning* terbagi menjadi tiga kategori: *Supervised Learning*, *Unsupervised Learning*, *Reinforcement Learning* (Somvanshi & Chavan, 2016).

1. *Supervised Learning*

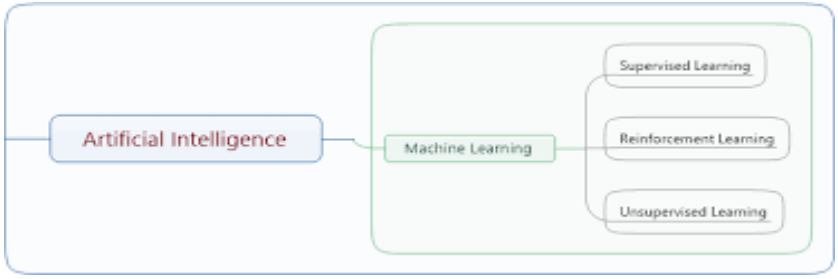
   Teknik yang digunakan oleh *Supervised Learning* adalah metode klasifikasi di mana kumpulan data sepenuhnya diberikan label untuk mengklasifikasikan kelas yang tidak dikenal (Homepage et al., 2019).

2. *Unsupervised Learning*

   Teknik yang digunakan *Unsupervised Learning* sering disebut cluster dikarenakan tidak ada kebutuhan untuk pemberian label dalam kumpulan data dan hasilnya tidak mengidentifikasi contoh di kelas yang telah ditentukan (Thupae, Isong, Gasela, & AbuMahfouz, 2018).

3. *Reinforcement Learning*

   Teknik yang digunakan *Reinforcement Learning* biasanya berada antara *Supervised Learning* dan *Unsupervised Learning* (Board, 2017), teknik ini bekerja dalam lingkungan yang dinamis di mana konsepnya harus menyelesaikan tujuan tanpa adanya pemberitahuan dari komputer secara eksplisit jika tujuan tersebut telah tercapai (Das & Nene, 2017).

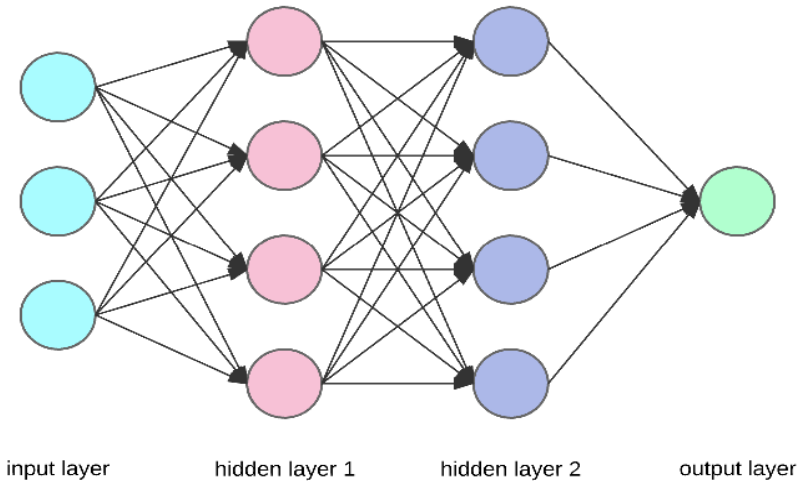


Gambar 3 Teknik *Reinforcement Learning*

2.2.7 Deep Learning

Deep Learning merupakan salat satu bidang *iranian car eruditeness* yang memanfaatkan banyak bed pengolahan informasi yang bekerja nonlinier untuk dapat melakukan ekstraksi fitur, pengenalan pola, dan klasifikasi. Menurut Goodfelow, *deep learning* merupakan sebuah pendekatan dalam melakukan penyelesaian masalah pada sistem pembelajaran komputer dengan memakai konsep hierarki. Konsep hierarki ini mampu mempelajari konsep yang kompleks iranian penggabungan konsep-konsep yang lebih sederhana. Jika digambarkan dalam sebuah *graf* bagaimana konsep tersebut dibangun di atas konsep yang lain, maka *graf* ini akan dalam dengan banyak bed hal ini yang

menjadi alasan mengapa disebut sebagai *Deep Learning* (pembelajaran mendalam) (Anggraini, n.d.).



Gambar 4 Graf Deep Learning

Beberapa algoritma termasuk dalam kategori *Deep Learning* antara lain:

a. *Convolutional Networks*

b. *Restricted Boltzmann Machine* (RBM)

c. *Deep Belief Networks* (DBN)

d. *Stacked Autocoders*

2.2.8 Artificial Neural Network (ANN)

Artificial Neural Network (Jaringan Syaraf Tiruan) adalah model non-linear yang kompleks, dibangun dari komponen yang secara individu berperilaku mirip dengan model regresi. Jaringan syaraf tiruan dapat divisualisasikan dengan grafik, dan beberapa sub-grafik beberapa memiliki perilaku yang sama dengan grafik sebelumnya. Meskipun struktur dari jaringan saraf secara eksplisit dirancang terlebih dahulu, pengolahan jaringan saraf ini tidak untuk menghasilkan hipotesis (berbagai neuron dan pengolahan lainnya terstruktur dalam jaringan) berkembang selama proses pembelajaran. Hal ini memungkinkan

neuron yang membentuk jaringan akan digunakan sebagai pemecahan masalah dari "program itu sendiri". (Yalidhan, 2018)

Artifial Neural Network menggunakan 3 jenis layer yang saling terhubung, yaitu:

a. *Input Layers*

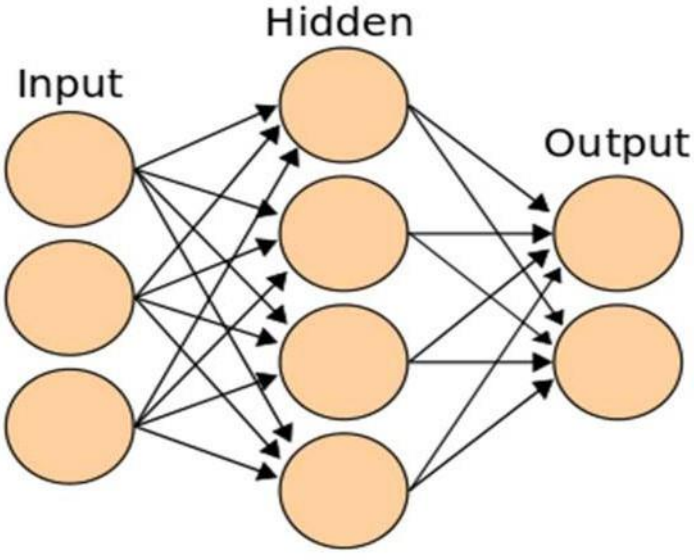
   Input layer adalah lapisan yang bertugas menerima input atau masukan langsung dari luar sistem. *Input layer* disesuaikan dengan jumlah input.

b. *Hidden Layer*

   Hidden layer adalah lapisan yang terletak diantara input layer dan output layer. *Hidden layer* terdiri dari neuron – neuron yang menerima data dari input layer.

c. *Output Layer*

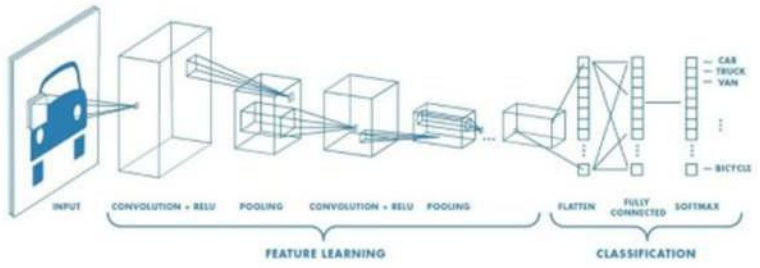
   Output layer adalah lapisan yang menghasilkan output akhir dari jaringan *artificial neural network*. Pada layer output ini jumlah neuron harus disesuakin dengan jumlah output yang diinginkan oleh system (Hadianto et al., 2019).



Gambar 5 *Artifial Neural Network* Layer

2.2.9 *Convolutional Neural Network* (CNN)

convolutional neural network adalah jenis jaringan saraf tiruan yang digunakan dalam pengenalan dan pemrosesan gambar. *convolutional neural network* meniru cara sel-sel saraf kita berkomunikasi dengan neuron yang saling berhubungan dan *convolutional neural network* memiliki arsitektur yang sama. Apa yang membuatnya unik dari jaringan saraf lain adalah operasi konvolusional yang menerapkan filter pada setiap bagian dari input sebelumnya untuk mengekstraksi pola dan *features maps*. Ahli statistik dan Peneliti telah mencari ide-ide jaringan saraf di abad ke-20 untuk mengerjakan Pengenalan Pola (Kholik, 2021).



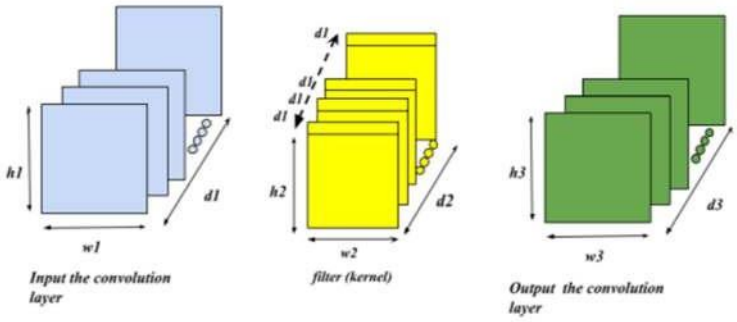
Gambar 6 *Convolutional Neural Network* (CNN)

Salah satu perkembangan terkenal adalah *Neocognitron* oleh Fukushima pada tahun 1980 yang memiliki sifat unik tidak terpengaruh oleh perubahan posisi, untuk pengenalan pola. Tetapi salah satu penelitian paling populer di bidang ini adalah pengembangan *LeNet-5*. Ini adalah salah satu dari *convolutional neural network* pertama yang digunakan di bank untuk membaca cek secara realtime, *LeNet-5* membaca lebih dari satu juta cek. Meskipun ada algoritma lain seperti *Support Vector Machine* yang akurasinya mendekati *LeNet-5*, namun banyak pendapat mengatakan bahwa kecepatan perhitungan *convolutional neural network* secara eksponensial lebih cepat daripada algoritma lain. Pada tahun 2010, untuk mendukung penelitian di bidang visi Komputer, *ImageNet* lahir. Imagenet adalah tempat penyimpanan dataset gambar yang sangat besar dan memiliki kompetisi yang terbuka setiap tahun untuk mempromosikan penelitian. Pada 2012, pemenang kompetisi *ImageNet* adalah model *Alexnet* oleh Alex Krizhevsky. *AlexNet* adalah model CNN yang mirip dengan *LeNet-5* tetapi lebih

signifikan dalam beberapa cara yang berdampak pada perkembangan *Artificial Intelligence*. *Convolutional neural network* juga memiliki kelemahan dalam proses pelatihan yang menyita waktu lama, Namun dengan didukungnya hardware sekarang ini sangat mudah untuk mengatasinya. Permasalahan yang lain adalah dengan kurangnya data pelatihan yang sesuai karena sulitnya mendapat dataset. Beberapa komponen utama yang ada dalam *convolutional neural network* yaitu *Convolution Layer, Pooling Layer, Fully Connected Layer dan Dropout.*

2.2.9.1 *Convolution Layer*

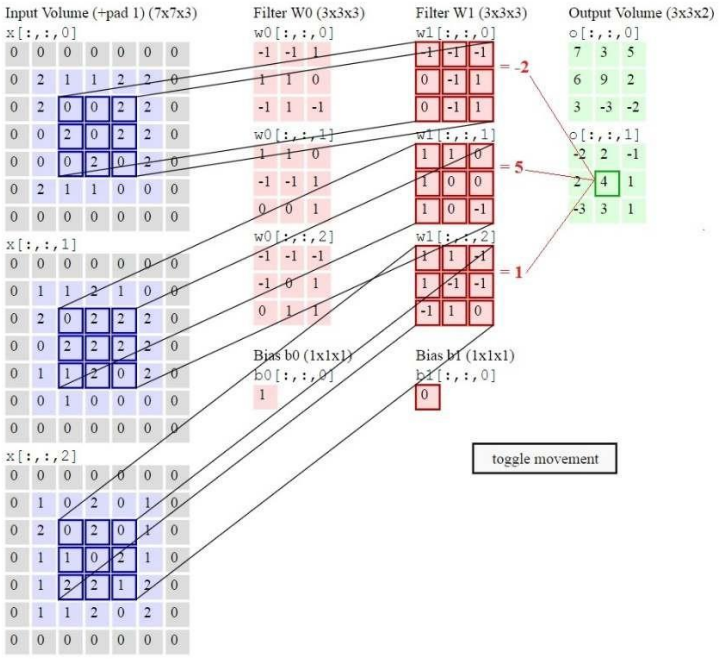
Blok bangunan utama *convolutional neural network* adalah *Convolution layer*. *Convolution* adalah operasi matematika untuk menggabungkan dua set informasi. *Convolution* diterapkan pada data input menggunakan filter *convolution* untuk menghasilkan *feature map.*



Gambar 7 *Convolution Layer*

Matrik biru pada gambar 2 adalah input dengan dimensi [$h1$ x w1 x $d1$]. Kernel (filter) adalah matriks dengan dimensi [$h2$ x w2 x $d2$], yang merupakan satu kuboid kuning dari beberapa kuboid (kernel) yang ditumpuk satu sama lain (dalam lapisan kernel). Setiap lapisan konvolusional ada beberapa kernel bertumpuk di atas satu sama lain, inilah yang membentuk matriks 3 dimensi kuning pada Gambar 2 yang merupakan dimensi

[$h2\ x$ w2 $x\ d2$], di mana d2 adalah jumlah kernel. Setiap kernel memiliki bias masing-masing yang merupakan jumlah skalar. Matriks berwarna hijau pada Gambar 2 adalah output yang memiliki dimensi [$h3\ x$ w3 $x\ d3$]. Kedalaman (d1) dari input dan satu kernel adalah sama. Kedalaman (d2) dari output sama dengan jumlah kernel (yaitu kedalaman matriks 3 dimensi berwarna hijau). Untuk setiap posisi kernel pada gambar, setiap angka dikalikan dengan angka yang sesuai pada matriks input (matriks biru) dan kemudian mereka semua dirangkum untuk nilai dalam posisi yang sesuai dalam matriks output (matriks hijau). Dengan d1 > 1, hal yang sama terjadi untuk masing-masing saluran dan kemudian mereka ditambahkan bersamasama kemudian disimpulkan dengan bias dari masing- masing filter dan ini membentuk nilai pada posisi yang sesuai dari matriks output.



Gambar 8 Matriks

2.2.9.2 Pooling Layer

Ada dua jenis tipe dari *Pooling* yaitu *Max Pooling* dan *Average Pooling*. *Max pooling* akan meringkas data masukan atau feature map dengan mencari nilai atau *value* terbesar dari *feature map* berdasarkan penggeseran *window pooling*. *Average pooling* akan menghitung nilai rata-rata dari setiap *feature map* pada setiap kali penggeseran *window pooling*. Contoh yang ditunjukkan pada Gambar 3 kernel ukuran $n \ x \ n$ ($3 \ x \ 3$) dipindahkan melintasi matriks dan untuk setiap posisi nilai maksimal diambil dan dimasukkan ke dalam posisi matriks keluaran yang sesuai, ini disebut *Max Pooling*. Dalam kasus *Average Pooling*, kernel dengan ukuran $n \ x \ n$ dipindahkan melintasi matriks dan untuk setiap posisi rata-rata diambil dari semua nilai dan dimasukkan ke dalam posisi yang sesuai dari matriks keluaran. Proses ini diulangi untuk setiap saluran dalam tensor input, hingga mendapatkan tensor output. Satu hal yang perlu diperhatikan adalah *pooling down* samples gambar dalam tinggi dan lebar, tetapi jumlah channels (kedalaman) tetap sama. Tujuan utama dari *Pooling Layer* adalah untuk mengurangi jumlah parameter dari tensor input sehingga dapat membantu mengurangi *overfitting*, ekstrak fitur representatif dari tensor input, mengurangi perhitungan dan dengan demikian membantu efisiensi.

2.2.9.3 Fully Connected Layer Output

Feature map yang dibuat dengan mengekstraksi fitur masih berupa array multidimensi, jadi *Feature map* perlu "diratakan" atau direformasi menjadi vektor sehingga dapat digunakan sebagai input vektor dari lapisan yang terhubung sepenuhnya. *Fully connected layer* adalah lapisan dimana semua neuron aktif pada lapisan sebelumnya terhubung dengan neuron pada lapisan berikutnya, mirip dengan jaringan syaraf tiruan. Setiap aktivitas pada level

sebelumnya harus diubah menjadi data satu arah sebelum terhubung ke semua neuron pada level yang terhubung penuh. Lapisan yang terhubung penuh biasanya digunakan dalam pendekatan MLP dan dimaksudkan untuk memproses data sehingga dapat dikategorikan. Perbedaan antara lapisan yang terhubung penuh dan lapisan konvolusi normal adalah bahwa neuron di lapisan konvolusi hanya terhubung ke area input tertentu. Di sisi lain, lapisan yang terhubung penuh memiliki neuron yang terhubung sepenuhnya. Namun, kedua lapisan tersebut membuat dot berfungsi, jadi tidak ada perbedaan besar dalam fungsi. *convolution layer* dengan ukuran kernel 1 x 1 melakukan fungsi yang sama dengan sebuah *fully connected layer* namun dengan tetap mempertahankan karakter spasial dari data. Hal tersebut membuat penggunaan *fully connected layer* pada *convolution neural network* sekarang tidak banyak dipakai.

2.2.10 *ResNet (Residual Network)*

Arsitektur *deep neural network* (DNN) buatan yang sangat sukses adalah *Residual Network* (ResNet) dan keluarganya (*ResNet-18, ResNet-50, ResNet-101, ResNet-152 juga ResNext*). Karakteristik utama dari arsitektur *ResNet* adalah menghindari menghilangnya masalah gradien. Masalah ini terjadi pada jaringan yang sangat dalam karena penurunan fungsi loss untuk menemukan bobot yang sesuai. Dalam *deep learning*, perkalian berulang menyebabkan gradien semakin kecil, hingga menghilang dengan bertambahnya jumlah lapisan. Selain itu, melatih jaringan saraf yang lebih dalam dengan banyak parameter membutuhkan learning rate yang signifikan (Wu et al. 2019). *ResNet* mengusulkan koneksi pintasan identitas yang dilewati dari beberapa lapisan dan menggunakan fungsi aktivasi lapisan sebelumnya. Ide seperti itu memungkinkan kita merancang DNN buatan dengan banyak lapisan dan tidak peduli dengan masalah gradien

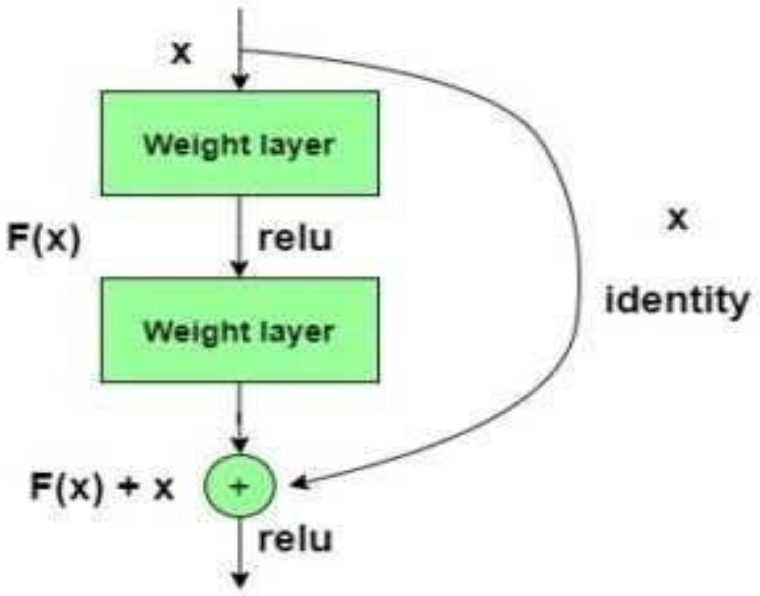
hilang. Untuk tujuan ini, dikembangkan arsitektur modular yang menumpuk blok sisa dengan pemetaan identitas. Pada blok residual, perhitungan Persamaan dilakukan:

$$x_{l+1} = x_l + F(x_l \cdot W_l)$$

xl dan xl+1 adalah input dan output dari blok residual ke-l. Wl adalah himpunan bobot, dan $F(xl \cdot \mathrm{W}l)$ adalah fungsi residual. Persamaan 5 dapat diwujudkan dengan *'shortcut connection'* dengan melakukan pemetaan identitas dan melewatkan satu atau lebih lapisan. Hal ini dilakukan berulang-ulang untuk setiap blok sisa L sehingga diperoleh:

$$x_L = x_l + \sum_{i=1}^{L-1} \quad F(x_i \cdot W_i)$$

Dari gradien fungsi loss, ketika bobot sangat kecil, gradien lapisan tidak hilang. Namun demikian, masih belum ada pengetahuan tentang posisi bagian objek (Falahkhi et al., n.d.).



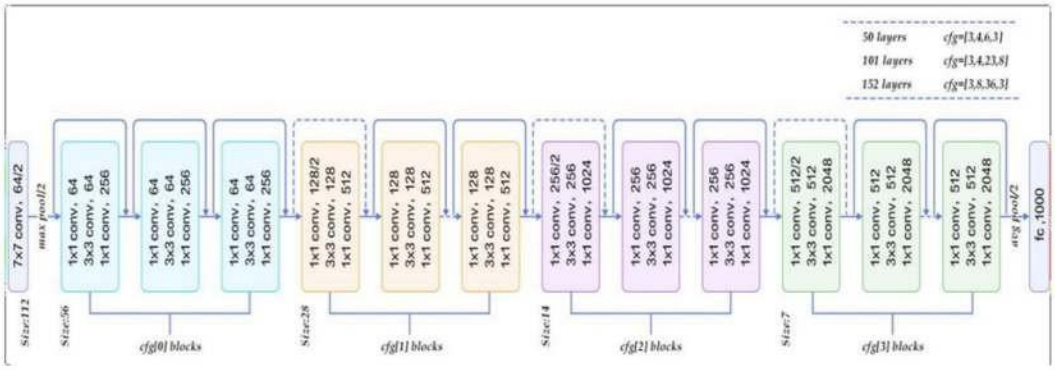
Gambar 9 *Residual Network* (ResNet)

Resnet memiliki berbagai macam jenis arsitektur, mulai dari 18, 34, 50, 101, sampai 152 layer (Hasan Mahmud & al Faraby, 2019). Alasan peneliti ingin menggunakan arsitektur ResNet ini karena ResNet

merupakan *pre trained* model atau model yang telah dilatih sebelumnya sehingga tidak memerlukan konfigurasi khusus untuk mengatur *layer* di dalamnya. Prinsip kerja ResNet adalah membangun jaringan yang lebih dalam dibandingkan dengan jaringan biasa lainnya dan secara bersamaan menemukan jumlah lapisan yang dioptimalkan untuk meniadakan masalah gradien yang hilang. Adapun manfaat dari penelitian ini adalah mengetahui seberapa akurat metode ResNet dalam melakukan klasifikasi jenis-jenis batik berdasarkan motifnya dan menjadi referensi bagi pembaca untuk melakukan penelitian yang berkaitan dengan klasifikasi gambar menggunakan metode ResNet. Hasil penelitian sebelumnya yaitu penelitian terhadap deteksi penyakit *Covid-19* menggunakan *Residual Network* (ResNet) menghasilkan akurasi yang tinggi dan evaluasi matrik yang baik walaupun data yang digunakan sedikit. Dengan akurasi yang tinggi dan hasil evaluasi matrik yang baik maka model tersebut dapat dipertimbangkan untuk diverifikasi sehingga bisa digunakan untuk mendeteksi penyakit *Covid-19.*

ResNet (Residual Network) merupakan jaringan residual yang memiliki jaringan yang dalam. Jaringan terdalam dari ResNet berjumlah 152 lapisan. Jaringan ini 8 kali lebih dalam dari jaringan VGG namun kompleksitasnya masih lebih rendah daripada jaringan VGG. Pada tahun 2015, Jaringan ini berhasil memenangkan juara pertama pada kompetisi ILSVRC (ImageNet Large Scale Visual Recognition Challenge) dan COCO dalam hal klasifikasi, deteksi dan segmentasi gambar pada dataset COCO dan ImageNet.

Berikut adalah arsitektur dan jumlah lapisan *ResNet*



Gambar 10 Arsitektur *ResNet*

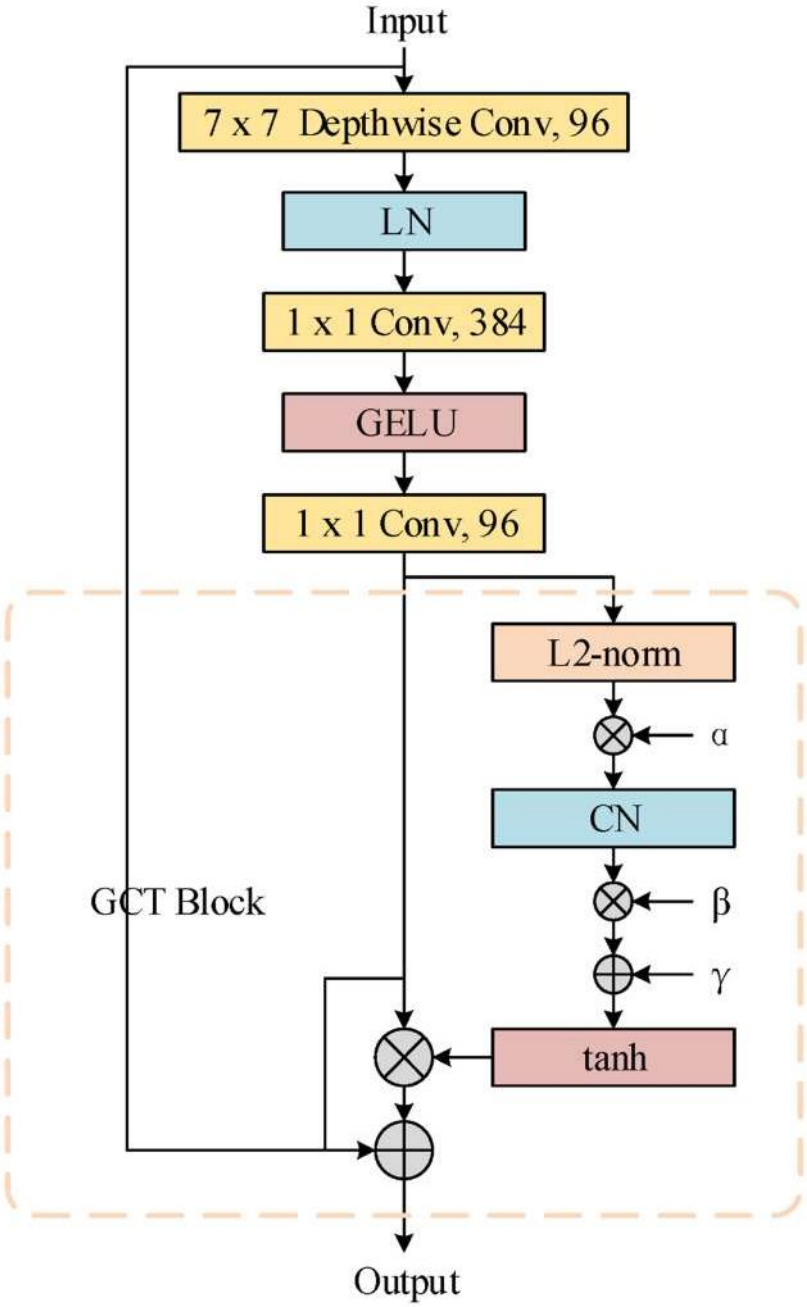
Pada ResNet 50 layer, untuk meningkatkan dimensi menggunakan projection shortcut pada pintasan identitas (identity shortcut) dan mengubah 2 layer block pada ResNet 34 layer dengan 3 layer bottleneck block. Sedangkan pada ResNet 101 dan 152 layer menggunakan lebih dari 3 layer block. Meskipun ResNet 152 layer kedalamannya meningkat secara signifikan, namun masih mempunyai kompleksitas yang lebih rendah dari ResNet50.

Model yang dibuat menggunakan hasil ekstraksi fitur gambar dari ResNet50. Nilai input model diambil dari vektor terakhir dari proses ekstraksi fitur. Model-2 menggunakan hasil ekstraksi fitur gambar dari ResNet101. Nilai input model diambil dari vektor terakhir hasil proses ekstraksi fitur.

2.2.11 *ConvNext*

ConvNeXt digunakan untuk menyesuaikan model *ResNet* klasik yang ada dan memperkenalkan beberapa ide dan teknologi terbaru dari model Swin Transformer ke dalam modul yang ada untuk meningkatkan kinerja klasifikasi model. Ini memiliki lima subversi, dan kami meningkatkan model ConvNeXt_L dengan lapisan jaringan yang lebih dalam. Jaringan backbone utama terdiri dari 4 tahap yang berbeda, setiap tahap terdiri dari beberapa blok, dan ConvNeXt digunakan untuk menyesuaikan rasio blok di setiap tahap menjadi 1:1:3:1. Itu menggantikan konvolusi 3 × 3 dengan konvolusi kedalaman 3 × 3 dan meningkatkan jumlah saluran dasar dari 64 menjadi 96. Kemudian, struktur leher botol terbalik diadopsi, sedangkan unit linier yang

diperbaiki (ReLU) dan normalisasi batch (BN) adalah digantikan oleh unit linier kesalahan Gaussian (GELU) dan normalisasi lapisan (LN). Terakhir, kernel konvolusi diperbesar menjadi 7×7 (Wei et al., 2023).



Gambar 11 *ConvNeXt*

2.2.12 *Python*

Python adalah bahasa pemrograman tingkat tinggi. *Python* dibuat oleh Guido van Rossum di Centrum Wiskunde & Informatica (CWI), Belanda dan pertama kali dirilis pada Tahun 1991. *Python* dapat

dipergunakan untuk proyek skala kecil ataupun besar. *Python* saat ini sudah mencapai versi 3.x dan dapat digunakan untuk berbagai kebutuhan seperti *web development, GUI development, scientific, software development, dan system administration*.

Python juga merupakan salah satu dari bahasa pemrograman tingkat tinggi yang bersifat interpreter, interaktif, object-oriented dan dapat beroperasi di hampir semua platform seperti keluarga *Linux*, *Windows*, *Mac*, dan platform lainnya. *Python* adalah salah satu bahasa pemrograman tingkat tinggi yang mudah dipelajari karena sintaks yang jelas dan elegan, yang dikombinasikan dengan penggunaan modul-modul yang mempunyai struktur data tingkat tinggi, efisien, dan siap langsung digunakan. *Source code* aplikasi dalam bahasa pemrograman *Python* biasanya akan dikompilasi menjadi format perantara yang dikenal sebagai *byte code* yang selanjutnya akan dieksekusi.

Gambar 12 Python

BAB III. METODOLOGI PENELITIAN

3.1. Waktu dan Tempat Penelitian

Pada bagian ini dituliskan tempat dilaksanakannya penelitian dan waktu untuk melakukan penelitian

3.2. Teknik Pengumpulan Data

Dalam penelitian ini pengumpulan data yang digunakan oleh peneliti adalah sumber data sekunder, karena data yang di peroleh adalah secara tidak langsung atau melalui sumber lain yang sudah tersedia datanya sebelum peneliti melakukan penelitian. Sumber yang digunakan untuk pengumpulan data melalui website http://www.kaggle.com/

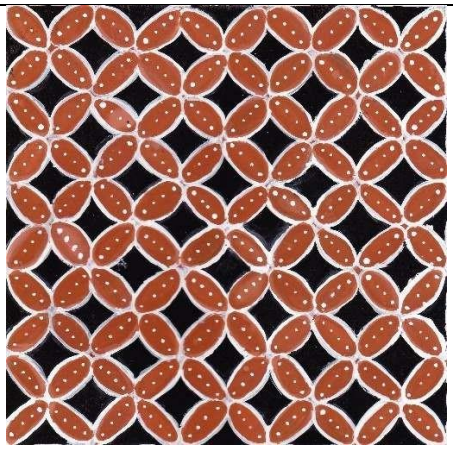
Dataset yang digunakan merupakan *dataset* public bersumber dari repository kaggle yaitu *Indonesian Batik Motif* yang dapat diakses di https://www.kaggle.com/datasets/dionisiusdh/indonesian-batik-motifs. *Dataset* ini terdiri dari 20 jenis batik dengan jumlah *dataset* sebanyak 2000 citra batik. Dari 20 kelas batik yang akan digunakan masing-masing terdapat 100 gambar batik di setiap kelasnya. Jadi total keseluruhan *dataset* yang akan digunakan untuk penelitian ini sekitar kurang lebih sekitar 2000 jenis gambar batik.

Berikut ini adalah tabel yang merupakan detail *dataset* yang akan digunakan pada penelitian ini.

Tabel 2 Contoh Jenis-Jenis Batik

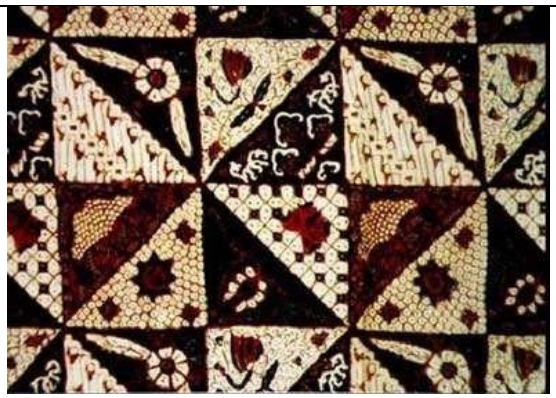
No.	Jenis Batik	Gambar
1.	Batik Bali	

2.	Batik Betawi	
3.	Batik Celup	
4.	Batik Cendrawasih	
5.	Batik Ceplok	

6.	Batik Ciamis	
7.	Batik Garutan	
8.	Batik Gentongan	
9.	Batik Kawung	

10.	Batik Keraton	
11.	Batik Lasem	
12.	Batik Megamendung	
13.	Batik Parang	

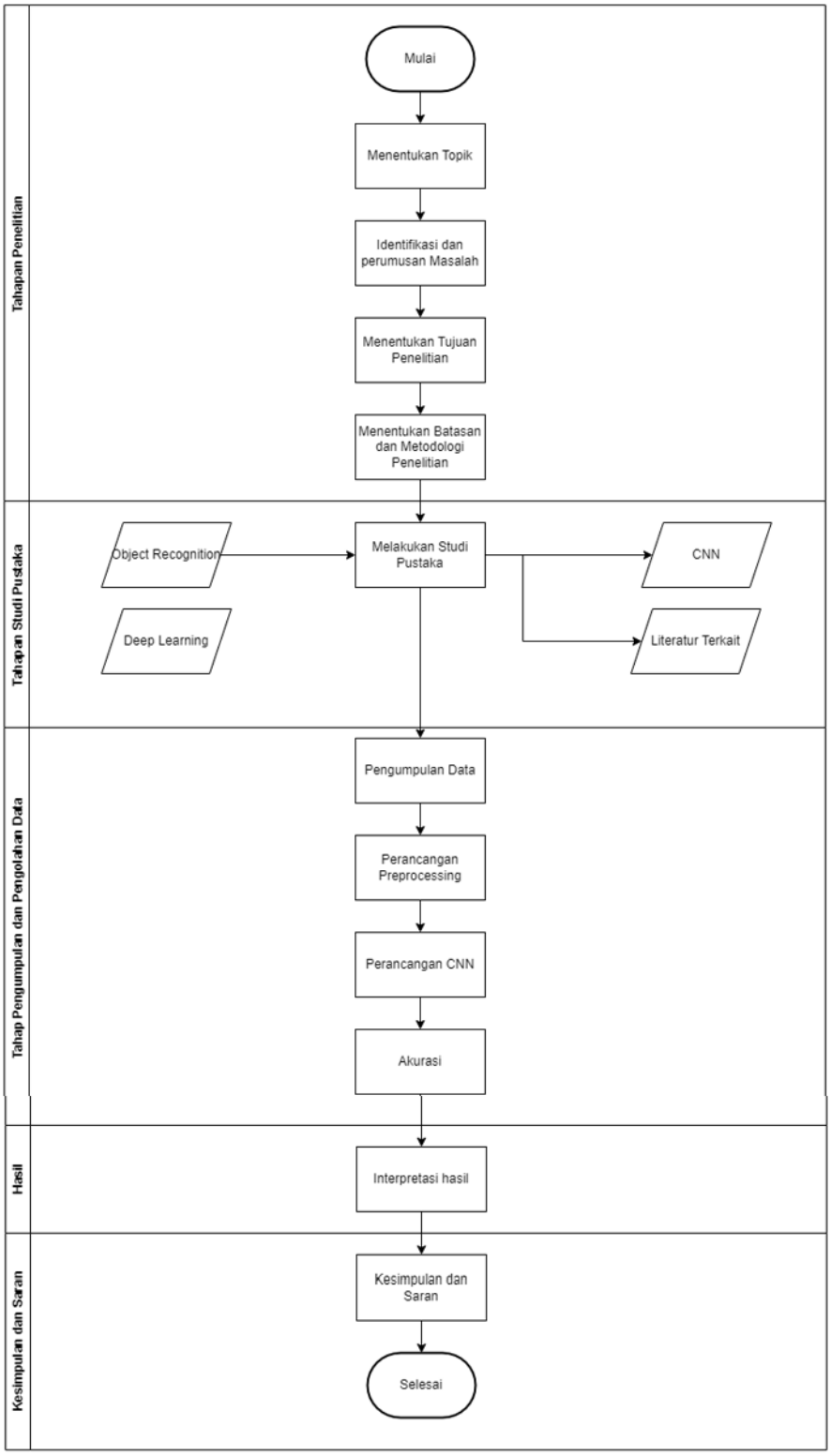
14.	Batik Pekalongan	
15.	Batik Priangan	
16.	Batik Sekar	
17.	Batik Sidoluhur	

18.	Batik Sidomukti	
19.	Batik Sogan	
20.	Batik Tambal	

Gambar 13 Macam Macam Batik

3.3 Tahapan Penelitian

Langkah atau tahapan yang dilakukan pada penelitian ini digambarkan melalui Gambar berikut ini:



Gambar 3. 1 Tahapan Penelitian

3.4 Teknik Pengolahan Data

Pada penelitian ini akan dibagi menjadi 4 tahap, yaitu tahap pengambilan data dan akuisisi citra, tahap preprocessing yaitu proses peningkatan kualitas citra. Tahap selanjutnya yaitu klasifikasi menggunakan CNN dengan arsitektur ResNet-50. Kemudian tahap pengujian sistem menggunakan skenario pengujian untuk memperoleh akurasi dalam menentukan jenis-jenis batik.

Gambar 14 Blok Diagram Tahapan Penelitian Secara Umum

3.4.1. Akuisisi Citra

Perancangan sistem diawali dengan proses akuisisi citra. Pada proses ini, dilakukan pengumpulan data untuk kebutuhan sistem sebagai data latih dan data uji. Kemudian dataset batik yang digunakan dalam penelitian ini didapatkan melalui website https://www.kaggle.com/datasets/dionisiusdh/indonesian-batik-motifs. *Dataset* ini terdiri dari 20 jenis batik dengan jumlah data sebanyak 2000 citra batik. Dan untuk setiap kelasnya terdapat sekitar 100 data jenis batik di dalamnya.

3.4.2. Pre-processing

Pada penelitian ini, sistem klasifikasi jenis batik ditambahkan tahap *preprocessing*, agar akurasi dapat lebih optimal. *Preprocessing* bertujuan untuk meningkatkan kualitas citra, sehingga memudahkan dan mempercepat kinerja sistem dalam mengenali motif pola jenis-jenis batik.

3.4.3. Pelatihan Model (Data Training)

Pelatihan model merupakan proses pelatihan mengenali objek dan mengklasifikasikannya sesuai dengan kelasnya. Penelitian ini menggunakan salah satu algoritma *deep learning* yaitu *Convolutional Neural Network* (CNN) dengan arsitektur ResNet-50.

3.4.4. Pengujian Sistem (Data Testing)

Untuk mendapatkan model sistem terbaik diperlukan parameter uji sebagai nilai pembanding untuk setiap model. Parameter performa yang digunakan dalam penelitian ini yaitu akurasi dan *loss*.

3.5 Uji Coba Sistem

3.5.1. Rancangan Pengujian

Pengujian ini dilakukan ntuk melakukan evaluasi terhadap model yang dihasilkan oleh CNN. Pengujian ini dilakukan dua tahap yaitu tahapan training dan testing. Tahap training adalah tahap dimana model CNN diuji dengan data latih yang sudah disediakan. Jumlah data latih yang disediakan sebanyak 2000 data gambar, dengan jumlah gambar perkelas sebanyak 100 gambar.

3.5.2. Perangkat Pengujian

Spesifikasi peralatan ujicoba yang digunakan peneliti terdapat pada tabel dibawah ini:

Tabel 3 Spesifikasi Peralatan Ujicoba

Processor	AMD Ryzen 5 3500U
RAM	8 GB
Sistem Operasi	Windows 10
Bahasa	Phyton
Training Platfoem	Google Colaboratory

3.5.3. Pengujian Model

a. Confusion Matrix

Confusion matrix merupakan metode yang digunakan untuk mengukur parameter performasi dari model klasifikasi dengan mencari akurasi, loss dan presisi. Pengukuran ini dibagi menjadi 4 istilah seperti tabel berikut ini

Tabel 4 Confusion Matrix

		Prediksi	
		True	False
Aktual	True	TP	FN
	False	FP	TN

Keterangan:

TP: *True Positif* didefinisikan sebagai positif yang diprediksi benar

TN: *True Negatif* didefinisikan sebagai negatif yang diprediksi benar

FP: *False Positif* didefinisikan sebagai negatif yang diprediksi dengan data positif

FN: *False Negatif* didefinisikan sebagai positif yang diprediksi dengan data negatif

b. Akurasi

Akurasi sistem adalah ketepatan sebuah sistem untuk mengenali data input agar output yang dihasilkan oleh sistem tersebut dapat sesuai dengan yang diharapkan. Berikut merupakan persaman akurasi

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN}$$

c. Loss

Loss adalah ketidaktepatan sistem untuk mengenali data input sehingga output yang dihasilkan tidak sesuai dengan yang diharapkan. Berikut merupakan persaman loss

$$Loss = \frac{FP + FN}{TP + TN + FP + FN}$$

d. Precision

Precision adalah tingkat ketepatan sistem dalam memberikan informasi agar sesuai dengan informasi yang diminta. Berikut merupakan persamaan precision

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

e. Recall

Recall adalah tingkat pengukuran sistem dalam menemukan informasi yang relevan. Berikut merupakan persamaan recall

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

f. F1-score

F1 Score atau F-Measure adalah perhitungan evaluasi dalam informasi retrieval yang mengkombinasikan antara recall dan precision. Berikut merupakan persamaan f1-score

$$F1 = \frac{2xPxR}{P + R}$$

BAB IV. ANALISIS DAN PERANCANGAN SISTEM

4.1. Deskripsi Sistem

Pelatihan yang dibangun dalam penelitian ini adalah penelitian tentang Implementasi *Image Classification* pada Jenis-Jenis Batik Menggunakan Algoritma *Convolutional Neural Network* dengan Model Arsitektur *ResNet*. Pelatihan ini digunakan untuk mengetahui akurasi terbaik dalam pelatihan model pada dataset batik karena banyaknya pola atau motif batik di Indonesia yang mengakibatkan sulitnya masyarakat untuk mengindentifikasi motif batik dengan akurat.

Dalam pelatihan ini dapat melihat akurasi terbaik pada dataset batik dengan menggunakan model arsitektur *Resnet* dengan melalui beberapa tahap seperti memasukkan dataset batik, melakukan perancangan *preprocessing*, melakukan pengumpulan data *training* dengan memuat dan memproses dataset untuk data *training*, melakukan pengumpulan data *validation* dengan memuat dan memproses dataset untuk data *validation*, proses *augmentasi*, menambahkan model *Resnet*, melakukan penambahan layer, menambahkan optimizer, dan melakukan pelatihan atau *testing*. Hasil dan tujuan dari pelatihan ini akan mendapatkan nilai akurasi yang baik.

4.2. Analisa Kebutuhan Sistem

Dalam membangun sistem pendukung keputusan ini, terdapat beberapa analisa pada sistem. Analisa yang dilakukan yakni analisa kebutuhan perangkat keras *(Hardware)* dan perangkat lunak *(Software)*. Berikut adalah spesifikasi dari analisa sistem :

A. Kebutuhan Perangkat Keras *(Hardware)*

Perangkat keras dibutuhkan untuk mendukung berjalannya program sesuai dengan ketentuan yang dibutuhkan. Berikut adalah spesifikasi perangkat yang digunakan dalam pembuatan sistem :

Tabel 5 Kebutuhan Perangkat Keras *(Hardware)*

No	Nama Perangkat Keras
1	AMD Ryzen 5 3500U with Radeon Vega Mobile Gfx 2.10 GHz
2	RAM 8GB

3	HDD 1TB

B. Kebutuhan Perangkat Lunak *(Software)*

Perangkat lunak dibutuhkan untuk membuat program sesuai dengan ketentuan yang dibutuhkan. Berikut adalah spesifikasi perangkat lunak yang digunakan dalam membuat sistem :

Tabel 6 Kebutuhan Perangkat Lunak (Software)

No	Nama Perangkat Lunak
1	Sistem Operasi Windows 10
2	Google Chrome

4.2.1. Kebutuhan Input

Dalam pelatihan image classification pada jenis-jenis batik ini membutuhkan data yang dibutuhkan untuk inputan citra pada pelatihan image classification. Inputan citra berupa dataset yang dapat diakses di https://www.kaggle.com/datasets/dionisiusdh/indonesian-batik-motifs. Dataset ini terdiri dari 20 jenis batik dengan jumlah data sebanyak 2000 citra batik. Selain itu, terdapat penambahan citra batik supaya menambahkan tingkat akurasi pada pelatihan ini.

4.2.2. Kebutuhan Proses

Setelah didapatkan dataset, kemudian dataset tersebut akan diproses supaya mendapatkan informasi yang dibutuhkan. Ada beberapa proses yang harus dilakukan, diantaranya:

4.2.1.1 Proses Augmentasi

Augmentasi data adalah suatu proses dalam pengolahan data gambar. Augmentasi menrupakan proses mengubah atau memodifikasi gambar sedemikian rupa sehingga computer akan medeteksi bahwa gambar yang diubah tersebut adalah gambar yang sama. Penggunaan proses augmentasi diharapkan dapat menaikkan performa model karena mesin akan berhasil mengenali lebih banyak objek dari bentuk serta pola yang beragam jenisnya. Berikut merupakan hasil augmentasi dataset batik untuk pelatihan ini.

Gambar 15 Hasil Proses Augmentasi

Berikut merupakan hasil augmentasi data dari citra dataset batik yang digunakan untuk pelatihan. Citra tersebut mengalami proses rescale, pembalikan citra secara horizontal dan mengalami rotasi acak.

```
train_datagen = ImageDataGenerator(
                    validation_split=0.3,
                    rescale=1./255,
                    horizontal_flip=True,
                    rotation_range=30)
```

Pada proses augmentasi ini, citra akan melakukan proses augmentasi guna memperkaya data baru untuk mendapatkan performa model yang baik. Citra akan mengalami beberapa proses augmentasi, seperti:

A. Validation Split

   Pada statement ini berfungsi untuk membagi data citra menjadi data training dan data validation. Proporsi dari argumen 0,3 (30%) dari jumlah dataset akan digunakan untuk data validasi. Sementara sisanya sebesar 0,7 (70%) akan digunakan untuk data training.

B. Rescale

   Pada statement ini berfungsi untuk menyalakan nilai piksel dalam citra menjadi rentang 0 hingga 1. Hal ini dilakukan dengan membagi setiap nilai piksel dengan 255. Prosedur ini dilakukan sebagai langkah normalisasi pada data citra.

C. Horizontal Flip

   Pada statement ini berfungsi untuk menerapkan atau mengaktifkan pembalikan horizontal secara acak pada citra dalam dataset.

D. Rotation Range

Pada statement ini berfungsi untuk mengatur rentang rotasi acak pada citra dalam dataset secara 30 derajat. Selama dalam pelatihan, citra pada dataset akan secara acak diputar dalam rentang -30 derajat hingga +30 derajat.

4.2.1.2 Proses Preprocessing

Proses preprocessing digunakan untuk meningkatkan kualitas gambar agar lebih baik daripada sebelumnya, dengan tujuan mempermudah pengenalan ciri-ciri pada gambar tersebut. Pada pelatihan kali ini citra akan melalui proses preprocessing dengan menggunakan filter Gaussian Noise.

```
      augmentasi = tf.keras.Sequential([

      tf.keras.layers.experimental.preprocessing.RandomFlip('hor
      izontal'),

      tf.keras.layers.experimental.preprocessing.RandomRotation(
      0.2),

      tf.keras.layers.experimental.preprocessing.RandomZoom(0.2)
      ,
          tf.keras.layers.GaussianNoise(0.2)
])
```

Pada proses preprocessing ini, citra akan melakukan proses prepocessing dari beberapa tahap preprocessing, seperti:

A. RandomFlip

Pada statement ini berfungsi untuk mengatur augmentasi berupa pembalikan acak secara horizontal pada citra. Argumen dari ‘horizontal’ menunjukkan citra dalam dataset akan mengalami pembalikan acak secara horizontal.

B. RandomRotation

Pada statement ini berfungsi untuk mengatur augmentasi berupa rotasi acak pada citra sebesar 0,2. Argumen dari 0,2 menunjukkan citra dalam dataset pelatikan akan secara acak di putar atau di rotasi sebesar 20%.

C. RandomZoom

Pada statement ini berfungsi untuk mengatur augmentasi berupa pembesaran acak pada citra sebesar 0,2. Argumen dari 0,2 menunjukkan citra dalam dataset pelatihan akan diperbesar sebesar 20%.

D. GaussianNoise

Pada statement ini berfunsi untuk mengaktifkan noise Gaussian secara acak pada citra dalam dataset. Argumen yang diberikan 0,2 merupakan standar deviasi dari distribusi Gaussian yang akan digunakan untuk menghasilkan noise.

4.2.1.3 Proses Penambahan Model Pelatihan

Pada proses ini, pelatihan akan menambahkan model pelatihan yang berfokus pada ResNet.

```
base_model = ResNet50(include_top=False,
weights='imagenet', input_shape=(224,224,3),
pooling='max')
```

Pada proses ini pelatihan akan mengaktifkan model ResNet50. Model akan menerima citra dengan ukuran 224 x 224 piksel dan 3 saluran warna (RGB)

4.2.1.4 Proses Penambahan Layer

Pada proses ini, pelatihan akan menambahkan beberapa layer guna meningkatkan hasil pelatihan dan mendapatkan hasil yang terbaik.

```
    model = tf.keras.Sequential([
        #augmentation,
        base_model,
        tf.keras.layers.Dropout(0.5),
        tf.keras.layers.Dense(128, activation='relu'),
        tf.keras.layers.Dropout(0.5),
        tf.keras.layers.Dense(512, activation='relu'),

    tf.keras.layers.Dense(len(train_generator.class_indices),
    activation='softmax')
])
```

Berikut merupakan beberapa penambahan layer yang akan digunakan untuk pelatihan ini:

A. Base_model

Pada argumen ini pelatihan akan memanggil objek base_model yang berisikan pengaktifan model ResNet.

B. Dropout

Pada argumen ini pelatihan akan menambahkan layer dropout dengan tingkat dropout sebesar 0,5. Dengan mengaktifkan layer dropout ini akan mencegah *overfitting* ketika pelatihan berlangsung. Penambahan layer Dropout ini dilakukan secara 2 kali dengan tingkat dropout yang sama.

C. Dense

Pada argumen ini pelatihan akan menambahkan layer dense yang akan dilakukan secara 1 kali. Untuk penambahan layer dense akan menggunakan 256 unit neuron dan menggunakan fungsi aktivasi ReLu.

Untuk penambahan layer dense yang terakhir menggunakan jumlah unit neuron yang sesuai dengan jumlah kelas dalam dataset pelatihan dengan statement `len(train_generator.class_indices)`. Selanjutnya pada layer dense ini juga menggunakan fungsi aktivasi softmax yang digunakan untuk menghasilkan probabilitas prediksi untuk setiap kelas.

4.2.1.5 Proses Penambahan Optimizer

Pada proses ini, pelatihan akan menambahkan optimizer. Optimizer ini digunakan mengoptimalkan parameter-parameter model neural network yang akan digunakan selama pelatihan.

```
model.compile(optimizer=RMSprop(learning_rate=0.0001),
loss='categorical_crossentropy',
metrics=['categorical_accuracy'])
```

Pada proses ini pelatihan akan membuat objek model.compile untuk menambahkan optimizer RMSprop yang akan digunakan selama proses pelatihan dengan menggunakan *learning rate* (tingkat pembelajaran) sebesar 0.0001.

4.2.1.6 Proses Pelatihan

Pada proses ini mesin akan melakukan pelatihan dengan proses-proses yang telah dirancang

```
history = model.fit(train_generator,
                    validation_data=validation_generator,
                    batch_size=64,
                    epochs=50,
                    verbose=1)
```

Pada proses pelatihan ini akan menggunakan data train yang diargumenkan dengan train_generator dan akan divalidasi menggunakan data validasi yang diargumenkan dengan validation_data. Selama pelatihan, ukuran batch yang digunakan sebesar 64 dan model akan dilatih selama 50 epoch (siklus pelatihan penuh) dengan verbosity level 1.

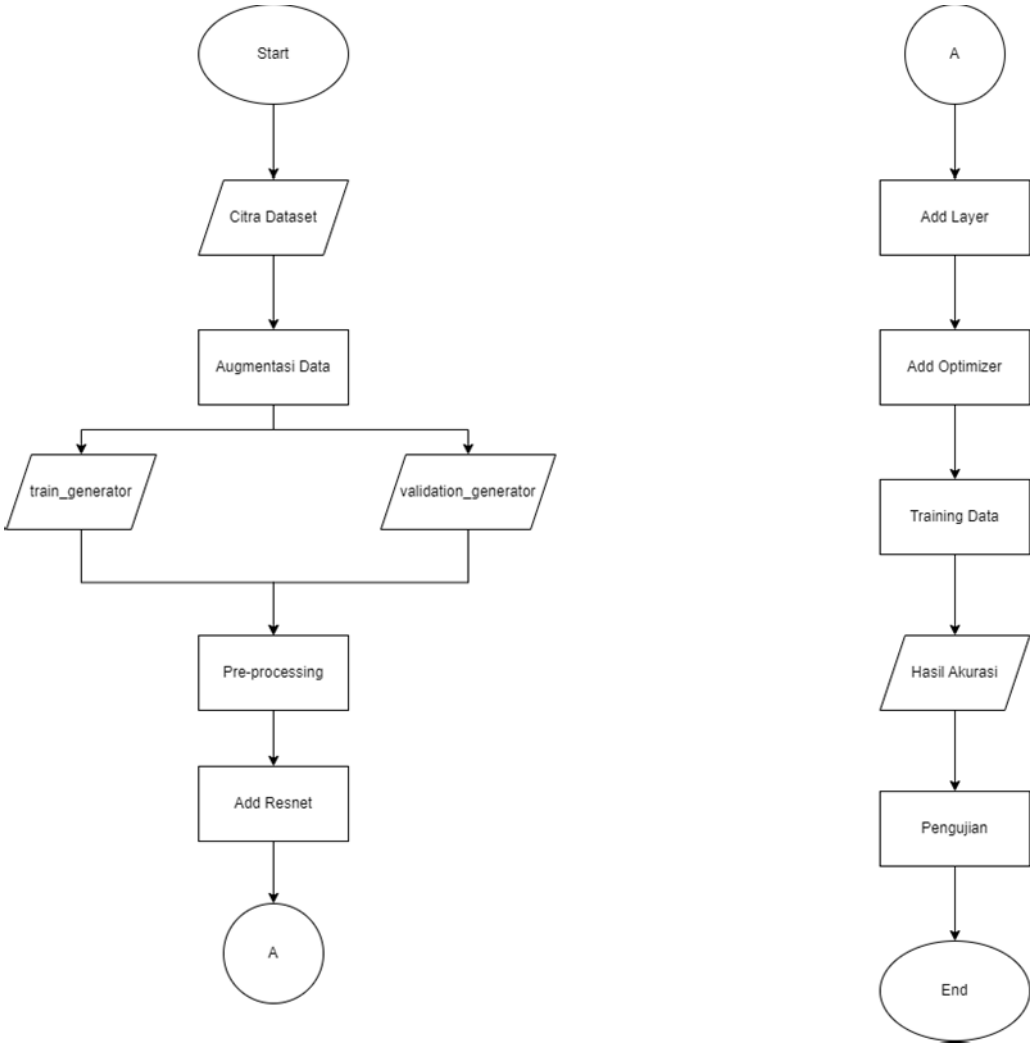
4.2.3. Kebutuhan Output

Output dari hasil input dataset dan beberapa proses seerta pelatihan dataset yaitu menghasilkan informasi berupa hasil akurasi yang diharapkan baik dan sesuai.

4.3. Perancangan Sistem

Pada tahap ini merupakan proses perancangan sistem untuk menghasilkan akurasi yang terbaik dari pelatihan yang telah dirancang.

4.2.4. Flowchart Sistem



Gambar 16 *Flowchart* Sistem

Berikut merupakan alur jalan sistem atau *flowchart* sistem pada penelitian tentang Implementasi *Image Classification* pada Jenis-Jenis Batik Menggunakan Algoritma *Convolutional Neural Network* dengan Model Arsitektur *ResNet*. Pada alur pertama mesin akan melakukan input data berupa dataset citra batik yang berisikan 20 class dengan jumlah citra sebanyak 2000 gambar.

Setelah mesin melakukan input dataset yang berisikan citra batik, mesin akan melakukan proses augmentasi dataset yang berisikan citra batik. Proses augmentasi ini memiliki fungsi untuk meningkatkan keanekaragaman dan jumlah data pelatihan tersedia. Dari proses augmentasi ini juga dapat meningkatkan performa model dan membantu untuk mencegah *overfitting*.

Setelah mesin melakukan proses augmentasi data, mesin akan membagi data citra menjadi 2, yaitu data train yang diargumenkan dengan train_genetaror dan data validasi yang diargumenkan dengan validation_generator. Proporsi untuk pembagian antara data train dengan data validasi sebanyak 70% untuk data train dan 30% untuk data validasi.

Setelah mesin melakukan proses pembagian data citra, mesin akan melakukan proses pre-processing. Pada tahap proses pre-processing kali ini, mesin akan menambahkan filter GaussianNoise senilai 0,2 yang akan digunakan untuk menghasilkan noise. Selain itu menambahkan filter Gaussian Noise, juga menambahkan Random Flip secara horizontal, Random Rotation, dan Random Zoom.

Setelah mesin melakukan proses pre-processing, mesin akan menambahkan *Resnet* didalam objek base_model. Pada Resnet kali ini akan menerima inputan citra dengan ukuran 224 x 224 piksel dan 3 saluran warna (RGB) yang diatur dalam input_shape.

Setelah mesin melakukan proses penambahan *Resnet*, mesin akan memperkenalkan lapisan tambahan yang bermanfaat untuk meningkatkan hasil pelatihan dan mencapai hasil yang optimal. Rancangan lapisan tambahan yang akan digunakan untuk pelatihan adalah sebagai berikut:

1. Lapisan pertama untuk *ResNet* yang diargumenkan dengan objek base_model.
2. Lapisan kedua untuk Dropout sebesar 0.5 untuk mencegah adanya *overfitting* ketika pelatihan berlangsung.
3. Lapisan ketiga untuk Dense dengan menggunakan 128 unit neuron dan menggunakan fungsi aktivasi ReLu.
4. Lapisan keempat untuk Dropout sebesar 0.5 untuk mencegah adanya *overfitting* ketika pelatihan berlangsung.
5. Lapisan kelima untuk Dense dengan menggunakan 512 unit neuron dan menggunakan fungsi aktivasi ReLu.
6. Lapisan keenam untuk Dense dengan menggunakan jumlah unit neuron yang disesuaikan dengan jumlah kelas yang dalam dataset pelatihan. Pada lapisan ini juga mengaktifkan fungsi softmax.

Setelah melakukan penambahan layers untuk pelatihan, tahap selanjutnya yaitu menambahkan *optimizer* yang digunakan mengoptimalkan parameter-parameter model neural network yang akan digunakan selama pelatihan.

Setelah mesin melakukan penambahan *optimizer*, mesin akan melakukan proses training data. Proses training data ini akan melibatkan seluruh proses yang telah dirancang. Pada proses training data ini akan dilakukan sebanyak 50 epoch dan nantinya akan menghasilkan sebuah nilai akurasi dari perancangan yang telah dirancang. Setelah mesin melakukan proses training dan menghasilkan hasil akurasi, akan melakukan pengujian dengan menggunakan *confussion matrix*.

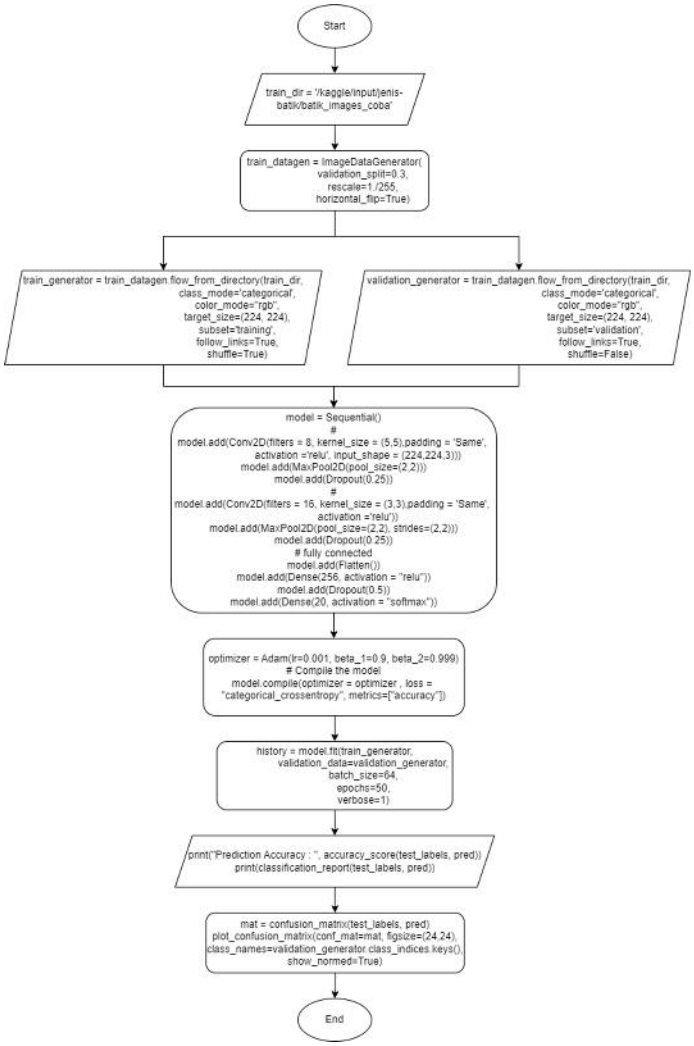
BAB V. IMPLEMENTASI DAN PENGUJIAN

5.1. Implementasi Uji Coba

Implementasi proses pelatihan yang ada pada pelatihan Implementasi *Image Classification* pada Jenis-Jenis Batik Menggunakan Algoritma *Convolutional Neural Network* dengan Model Arsitektur *ResNet* dari beberapa potongan kode program sebagai berikut.

5.1.1. Skenario 1

Uji coba ini dilakukan dengan melakukan proses *training* dengan menggunakan algoritma *Convolutional Neural Network* tanpa menggunakan *ResNet*.



Gambar 17 *Flowchart Code* Skenario 1

Berikut merupakan alur jalan code atau *flowchart* code pada penelitian tentang Implementasi *Image Classification* pada Jenis-Jenis Batik Menggunakan Algoritma *Convolutional Neural Network* dengan Model Arsitektur *ResNet* untuk skenario 1. Pada alur pertama mesin akan melakukan input data berupa dataset citra batik yang berisikan 20 class dengan jumlah citra sebanyak 2000 gambar. Inputan dataset citra batik ini akan menginisialisasi objek dengan nama train_dir.

```
train_dir = '/kaggle/input/jenis-batik/batik_images_coba'
```

Setelah mesin melakukan input dataset yang berisikan citra batik, mesin akan melakukan proses augmentasi dataset yang berisikan citra batik. Proses augmentasi ini memiliki fungsi untuk meningkatkan keanekaragaman dan jumlah data pelatihan tersedia. Dari proses augmentasi ini juga dapat meningkatkan performa model dan membantu untuk mencegah *overfitting*.

```
train_datagen = ImageDataGenerator(
                    validation_split=0.3,
                    rescale=1./255,
                    horizontal_flip=True,
                    rotation_range=30)
```

Pada proses augmentasi data ini akan menginisialisasi objek dengan nama train_datagen. Proses augmentasi data ini melibatkan kelas ImageDataGenerator dalam library TensorFlow. ImageDataGenerator merupakan alat yang berguna untuk mempersiapkan data gambar untuk pelatihan model jaringan saraf. Pada proses augmentasi ini, akan melakukan beberapa perlakuan untuk data seperti:

1. validation split yang memiliki fungsi untuk membagi data training dan data validation yang digunakan untuk pelatihan. Hasil dari pembagian data ini adalah 30% untuk data validasi dan 70% untuk data training.
2. rescale yang memiliki fungsi untuk menormalisasi data citra dengan membagi setiap nilai piksel dengan 255.
3. horizontal_flip yang memiliki fungsi untuk mengaktifan pembalikan citra pada dataset secara horizontal.
4. rotation range yang memiliki fungsi untuk mengaktifkan rotasi citra pada dataset senilai 30 derajat.

Setelah mesin melakukan proses augmentasi data, mesin akan membagi data citra menjadi 2, yaitu data train yang diargumenkan dengan train_genetaror

dan data validasi yang diargumenkan dengan validation_generator. Proporsi untuk pembagian antara data train dengan data validasi sebanyak 70% untuk data train dan 30% untuk data validasi.

```
train_generator = train_datagen.flow_from_directory(
          train_dir,
          class_mode='categorical',
          color_mode="rgb",
          target_size=(224, 224),
          subset='training',
          follow_links=True,
          shuffle=True)
```

```
validation_generator = train_datagen.flow_from_directory(
          train_dir,
          class_mode='categorical',
          color_mode="rgb",
          target_size=(224, 224),
          subset='validation',
          follow_links=True,
          shuffle=False)
```

Setelah melakukan pembagian data antara data training dengan data validation, tahap selanjutnya adalah melakukan penambahan rancangan layer sesuai dengan arsitektur *Convolutional Neural Network*.

```
model = Sequential()
#
model.add(Conv2D(filters = 8, kernel_size = (5,5),padding = 'Same',
                 activation ='relu', input_shape = (224,224,3)))
model.add(MaxPool2D(pool_size=(2,2)))
model.add(Dropout(0.25))
#
model.add(Conv2D(filters = 16, kernel_size = (3,3),padding = 'Same',
                 activation ='relu'))
model.add(MaxPool2D(pool_size=(2,2), strides=(2,2)))
model.add(Dropout(0.25))
# fully connected
model.add(Flatten())
model.add(Dense(256, activation = "relu"))
model.add(Dropout(0.5))
model.add(Dense(20, activation = "softmax"))
```

Berikut merupakan rancangan layer untuk uji coba pelatihan menggunakan arsitektur Convolutional Neural Network

A. Conv2D

Conv2D merupakan lapisan konvolusi pertama pada uji coba pelatihan kali ini. Pada lapisan ini menggunakan 8 filter konvolusi dengan ukuran kernel 5x5. Statemet padding = ‘same’ digunakan untuk mempertahankan dimensi gambar input. Pada lapisan ini juga mengaktifkan fungsi aktivasi ReLu dan memiliki inputan gambar sebesar 224x224 piksel dan 3 saluran gambar (RGB).

B. MaxPool2D

MaxPool2D merupakan lapisan pooling pertama. Lapisan ini menggunakan *max pooling* dengan ukuran pool 2x2 yang memiliki fungsi untuk mengurangi ukuran representasi fitur.

C. Dropout

Dropout kali ini merupakan lapisan dropout pertama. Lapisan dropout digunakan untuk menghindari *overfitting*. Pada lapisan pertama dropout menggunakan tingkat dropout sebesar 0.25

D. Conv2D

Conv2D kali ini merupakan lapisan konvolosi yang kedua. Pada lapisan ini menggunakan 16 filter konvolusi dengan ukuran kernel 3x3.

E. MaxPool2D

MaxPool2D kali ini merupakan lapisan *max pooling* yang kedua. Lapisan ini menggunakan *max pooling* dengan ukuran pool 2x2 dan pergeseran (*strides*) 2x2 untuk mengurangi representasi fitur

F. Dropout

Droput kali ini merupakan lapisan dropout kedua. Pada lapisan kedua dropout menggunakan tingkat dropout sebesar 0.25

G. Flatten

Flatten merupakan lapisan *fully connected* yang digunakan untuk mengubah representasi matriks 2D yang dihasilkan oleh lapisan konvolusi sebelumnya menjadi vektor 1D

H. Dense

Dense merupakan lapisan output yang terhubung penuh (*fully connected layer*) dengan memiliki 256 unit neuron serta mengaktifkan fungsi aktivasi ReLu yang memungkinkan keluaran positif untuk melalui neuron dan mematikan keluaran negatif.

I. Dropout

Droput kali ini merupakan lapisan dropout ketiga. Pada lapisan ketiga dropout mennggunakan tingkat dropout sebesar 0.5

J. Dense

Lapisan output ini merupakan lapisan terhubung penuh (*fully connected layer*) dengan 20 unit neuron yang sesuai dengan jumlah kelas yang diharapkan. Fungsi aktivasi *softmax* digunakan untuk menghasilkan probabilitas prediksi untuk setiap kelas.

Setelah mesin melakukan perancangan layer untuk *Convolutional Neural Network*, proses selanjutnya yaitu penambahan optimizer dengan menggunakan optimizer Adam dan menggunakan *learning rate* sebesar 0.001.

```
optimizer = Adam(lr=0.001, beta_1=0.9, beta_2=0.999)
```

Setelah mesin melakukan penambahan *optimizer*, maka tahap selanjutnya adalah tahap pelatihan (*training*). Proses training data ini akan melibatkan seluruh proses yang telah dirancang. Pada proses training data ini akan dilakukan sebanyak 50 epoch.

```
history = model.fit(train_generator,
                    validation_data=validation_generator,
                    batch_size=64,
                    epochs=50,
                    verbose=1)
```

Setelah mesin melakukan proses training, mesin akan mencetak hasil dari proses training data. Hasil dari proses training meliputi: grafik training, akurasi, precision, recall dan f1-score

1. Hasil proses training menggunakan grafik training

```
import matplotlib.pyplot as plt

metrics = history.history

plt.plot(history.epoch,             metrics['categorical_accuracy'],
metrics['val_categorical_accuracy'])

plt.legend(['categorical_accuracy', 'val_categorical_accuracy'])

plt.show()



plt.plot(history.epoch, metrics['loss'], metrics['val_loss'])

plt.legend(['loss', 'val_loss'])

plt.show()
```

2. Hasil proses training menggunakan akurasi, precision, recall dan f1-score

```
test_labels = validation_generator.classes

predictions = model.predict(validation_generator)

pred = np.argmax(predictions, axis=1)



from sklearn.metrics import accuracy_score

print("Prediction Accuracy : ", accuracy_score(test_labels, pred))



from sklearn.metrics import classification_report

print(classification_report(test_labels, pred))
```

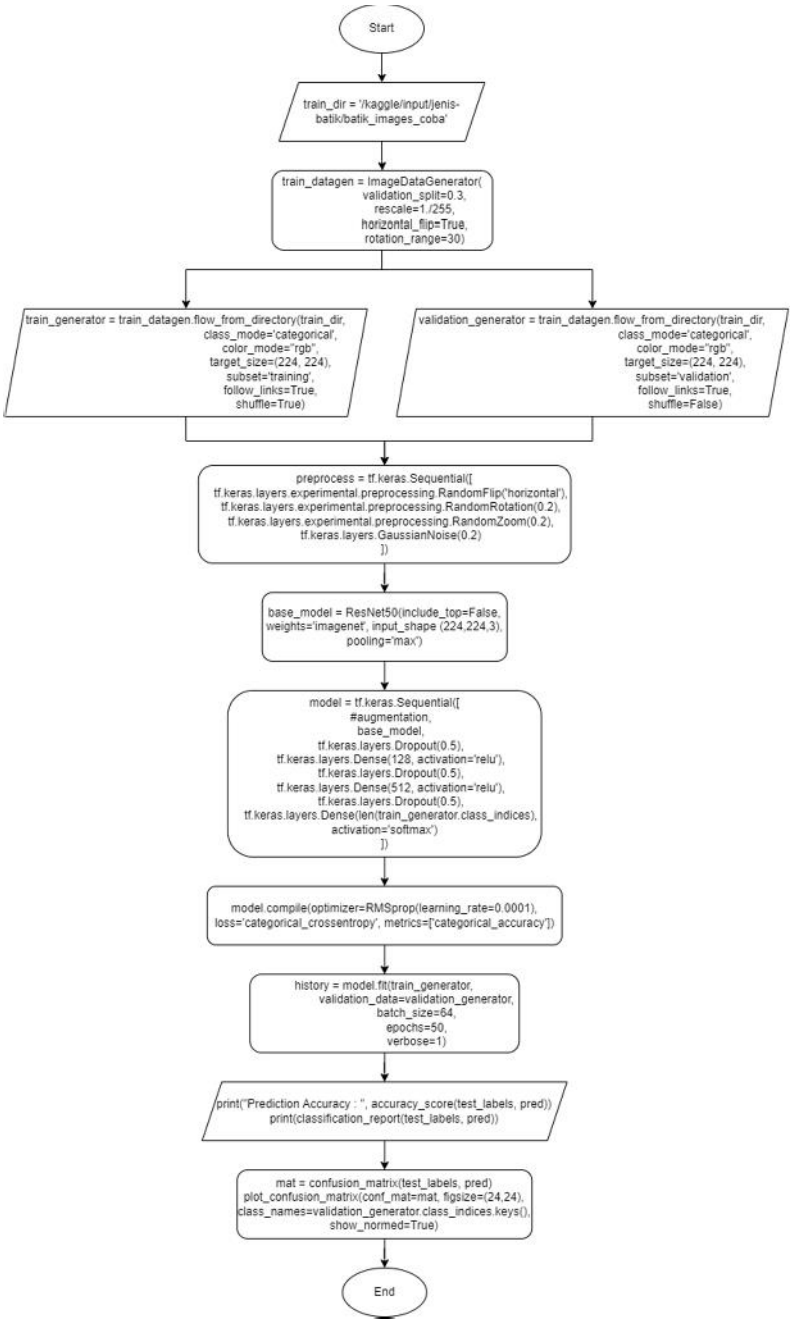
Proses selanjutnya dari proses training citra batik adalah tahap pengujian. Uji coba pelatihan kali ini akan menggunakan metode *confusion matrix*.

```
from mlxtend.plotting import plot_confusion_matrix

from sklearn.metrics import confusion_matrix



mat = confusion_matrix(test_labels, pred)
```

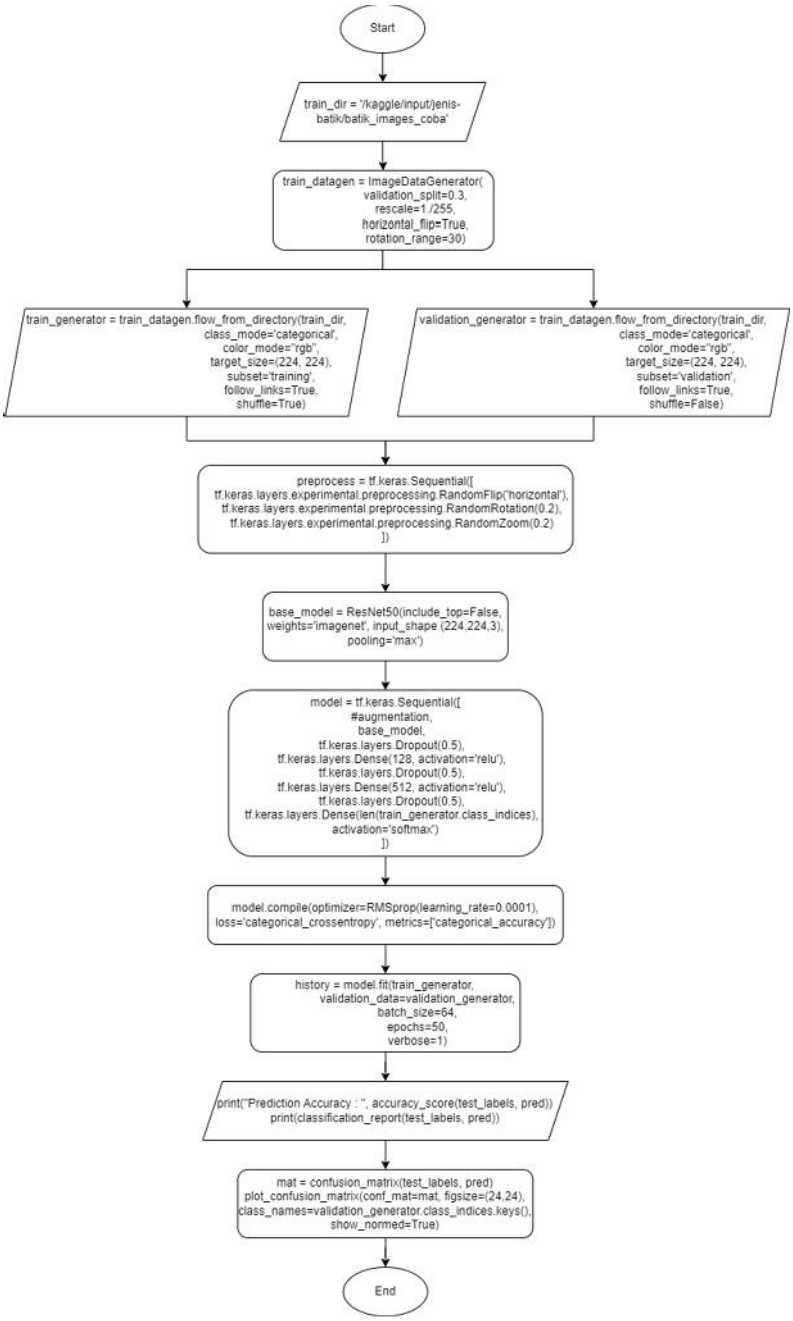
```
plot_confusion_matrix(conf_mat=mat,              figsize=(24,24),
class_names=validation_generator.class_indices.keys(),
show_normed=True)
```

5.1.2. Skenario 2

Uji coba ini dilakukan dengan membandingkan model dengan menggunakan data *preprocessing* dengan model tanpa *preprocessing*. *Preprocessing* yang digunakan pada uji coba ini yaitu menggunakan filter *GaussianNoise* dan akan melakukan pelatihan sebanyak 50 *epoch*.



Gambar 18 *Flowchart Code* Skenario 2 dengan data *preprocessing*



Gambar 19 *Flowchart Code* Skenario 2 tanpa data *preprocessing*

Berikut merupakan alur jalan code atau *flowchart* code pada penelitian tentang Implementasi *Image Classification* pada Jenis-Jenis Batik Menggunakan Algoritma *Convolutional Neural Network* dengan Model Arsitektur *ResNet* untuk skenario 2. Pada alur pertama mesin akan melakukan input data berupa dataset citra batik yang berisikan 20 class dengan jumlah citra sebanyak 2000 gambar. Inputan dataset citra batik ini akan menginisialisasi objek dengan nama train_dir.

```
train_dir = '/kaggle/input/jenis-batik/batik_images_coba'
```

Setelah mesin melakukan input dataset yang berisikan citra batik, mesin akan melakukan proses augmentasi dataset yang berisikan citra batik. Proses augmentasi ini memiliki fungsi untuk meningkatkan keanekaragaman dan jumlah data pelatihan tersedia. Dari proses augmentasi ini juga dapat meningkatkan performa model dan membantu untuk mencegah *overfitting*.

```
train_datagen = ImageDataGenerator(
                    validation_split=0.3,
                    rescale=1./255,
                    horizontal_flip=True,
                    rotation_range=30)
```

Pada proses augmentasi data ini akan menginisialisasi objek dengan nama train_datagen. Proses augmentasi data ini melibatkan kelas ImageDataGenerator dalam library TensorFlow. ImageDataGenerator merupakan alat yang berguna untuk mempersiapkan data gambar untuk pelatihan model jaringan saraf. Pada proses augmentasi ini, akan melakukan beberapa perlakuan untuk data seperti:

1. validation split yang memiliki fungsi untuk membagi data training dan data validation yang digunakan untuk pelatihan. Hasil dari pembagian data ini adalah 30% untuk data validasi dan 70% untuk data training.
2. rescale yang memiliki fungsi untuk menormalisasi data citra dengan membagi setiap nilai piksel dengan 255.
3. horizontal_flip yang memiliki fungsi untuk mengaktifan pembalikan citra pada datset secara horizontal.
4. rotation_range yang memiliki fungsi untuk mengaktifkan rotasi citra pada dataset senilai 30 derajat.

Setelah mesin melakukan proses augmentasi data, mesin akan membagi data citra menjadi 2, yaitu data train yang diargumenkan dengan train_genetaror dan data validasi yang diargumenkan dengan validation_generator. Proporsi untuk pembagian antara data train dengan data validasi sebanyak 70% untuk data train dan 30% untuk data validasi.

```
train_generator = train_datagen.flow_from_directory(
          train_dir,
          class_mode='categorical',
```

```
          color_mode="rgb",
          target_size=(224, 224),
          subset='training',
          follow_links=True,
          shuffle=True)
validation_generator = train_datagen.flow_from_directory(
          train_dir,
          class_mode='categorical',
          color_mode="rgb",
          target_size=(224, 224),
          subset='validation',
          follow_links=True,
          shuffle=False)
```

Setelah mesin melakukan proses pembagian data citra, mesin akan melakukan proses pre-processing. Pada skenario uji coba kali ini, mesin akan menambahkan data preprocessing filter GaussianNoise senilai 0,2 yang akan digunakan untuk menghasilkan noise dan tidak menggunakan data preprocessing.

1. Model tanpa menggunakan data *preprocessing*

```
augmentation = tf.keras.Sequential([
tf.keras.layers.experimental.preprocessing.RandomFlip('horizonta
l'),
tf.keras.layers.experimental.preprocessing.RandomRotation(0.2),
tf.keras.layers.experimental.preprocessing.RandomZoom(0.2)
])
```

2. Model menggunakan data *preprocessing* filter GaussianNoise senilai 0.2

```
preprocess = tf.keras.Sequential([
tf.keras.layers.experimental.preprocessing.RandomFlip('horizonta
l'),
tf.keras.layers.experimental.preprocessing.RandomRotation(0.2),
tf.keras.layers.experimental.preprocessing.RandomZoom(0.2),
tf.keras.layers.GaussianNoise(0.2)
])
```

Setelah mesin melakukan proses pre-processing, mesin akan menambahkan *ResNet* didalam objek base_model. Pada ResNet kali ini akan menerima inputan citra dengan ukuran 224 x 224 piksel dan 3 saluran warna (RGB) yang diatur dalam input_shape.

```
base_model = ResNet50(include_top=False, weights='imagenet',
input_shape=(224,224,3), pooling='max')
```

Setelah mesin melakukan penambahan layer untuk ResNet, mesin akan menambahkan racangan layer seperti dengan potongan code tersebut.

```
        model = tf.keras.Sequential([
            #augmentation,
            base_model,
            tf.keras.layers.Dense(256, activation='relu'),
            tf.keras.layers.Dense(len(train_generator.class_indices),
        activation='softmax')
])
```

Setelah mesin melakukan penambahan layer sesuai dengan rancangan, tahap selanjutnya adalah penambahan *optimizer*. Pada pelatihan kali ini menggunakan RMSprop dengan *learning rate* sebesar 0.0001.

```
model.compile(optimizer=RMSprop(learning_rate=0.0001),
loss='categorical_crossentropy',
metrics=['categorical_accuracy'])
```

Setelah mesin melakukan penambahan *optimizer*, maka tahap selanjutnya adalah tahap pelatihan (*training*). Proses training data ini akan melibatkan seluruh proses yang telah dirancang. Pada proses training data ini akan dilakukan sebanyak 50 epoch.

```
        history = model.fit(train_generator,
                            validation_data=validation_generator,
                            batch_size=64,
                            epochs=50,
                           verbose=1)
```

Setelah mesin melakukan proses training, mesin akan mencetak hasil dari proses training data. Hasil dari proses training meliputi: grafik training, akurasi, precision, recall dan f1-score.

1. Hasil proses training menggunakan grafik training

```
import matplotlib.pyplot as plt

metrics = history.history

plt.plot(history.epoch,              metrics['categorical_accuracy'],
metrics['val_categorical_accuracy'])

plt.legend(['categorical_accuracy', 'val_categorical_accuracy'])

plt.show()
```

```
plt.plot(history.epoch, metrics['loss'], metrics['val_loss'])
plt.legend(['loss', 'val_loss'])
plt.show()
```

2. Hasil proses training menggunakan akurasi, precision, recall dan f1-score

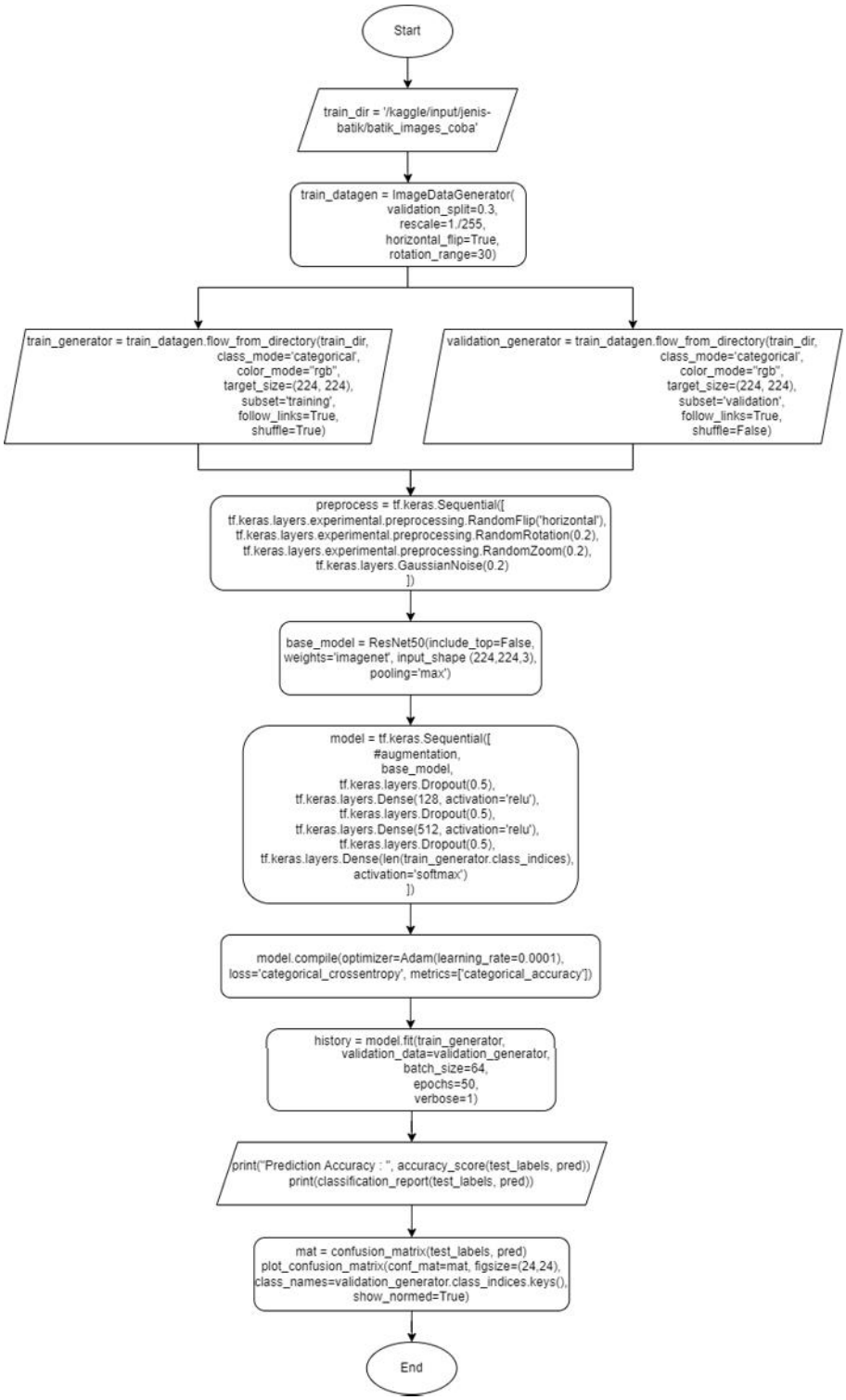
```
test_labels = validation_generator.classes
predictions = model.predict(validation_generator)
pred = np.argmax(predictions, axis=1)

from sklearn.metrics import accuracy_score
print("Prediction Accuracy : ", accuracy_score(test_labels, pred))

from sklearn.metrics import classification_report
print(classification_report(test_labels, pred))
```

Proses selanjutnya dari proses training citra batik adalah tahap pengujian. Uji coba pelatihan kali ini akan menggunakan metode *confusion matrix*.

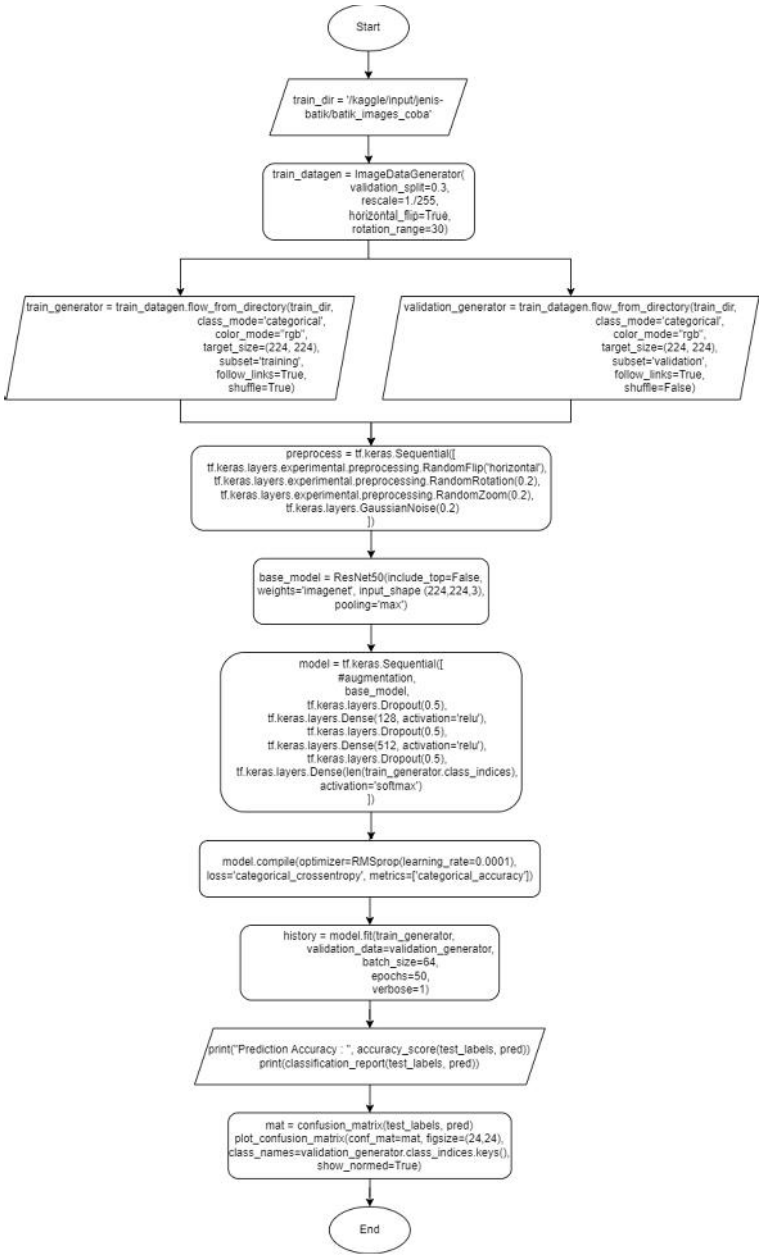
```
from mlxtend.plotting import plot_confusion_matrix
from sklearn.metrics import confusion matrix

mat = confusion_matrix(test_labels, pred)
plot_confusion_matrix(conf_mat=mat,              figsize=(24,24),
class_names=validation_generator.class_indices.keys(),
show_normed=True)
```

5.1.3. Skenario 3

Uji coba ini dilakukan dengan membandingkan model dengan menggunakan *optimizer* Adam dan RMSprop dengan *learning rate* sebesar 0.0001 dan melakukan pelatihan sebanyak 50 epoch.



Gambar 20 *Flowchart Code* Skenario 3 dengan *optimizer* Adam *learning rate* 0.0001



Gambar 21 *Flowchart Code* Skenario 3 dengan *optimizer* RMSprop *learning rate* 0.0001

Berikut merupakan alur jalan code atau *flowchart* code pada penelitian tentang Implementasi *Image Classification* pada Jenis-Jenis Batik Menggunakan Algoritma *Convolutional Neural Network* dengan Model Arsitektur *ResNet* untuk skenario 3. Pada alur pertama mesin akan melakukan input data berupa dataset citra batik yang berisikan 20 class dengan jumlah citra sebanyak 2000 gambar. Inputan dataset citra batik ini akan menginisialisasi objek dengan nama train_dir.

```
train_dir = '/kaggle/input/jenis-batik/batik_images_coba'
```

Setelah mesin melakukan input dataset yang berisikan citra batik, mesin akan melakukan proses augmentasi dataset yang berisikan citra batik. Proses augmentasi ini memiliki fungsi untuk meningkatkan keanekaragaman dan jumlah data pelatihan tersedia. Dari proses augmentasi ini juga dapat meningkatkan performa model dan membantu untuk mencegah *overfitting*.

```
train_datagen = ImageDataGenerator(
                    validation_split=0.3,
                    rescale=1./255,
                    horizontal_flip=True,
                    rotation_range=30)
```

Setelah mesin melakukan proses augmentasi data, mesin akan membagi data citra menjadi 2, yaitu data train yang diargumenkan dengan train_genetaror dan data validasi yang diargumenkan dengan validation_generator. Proporsi untuk pembagian antara data train dengan data validasi sebanyak 70% untuk data train dan 30% untuk data validasi.

```
train_generator = train_datagen.flow_from_directory(
          train_dir,
          class_mode='categorical',
          color_mode="rgb",
          target_size=(224, 224),
          subset='training',
          follow_links=True,
          shuffle=True)
```

```
validation_generator = train_datagen.flow_from_directory(
          train_dir,
          class_mode='categorical',
          color_mode="rgb",
          target_size=(224, 224),
          subset='validation',
          follow_links=True,
          shuffle=False)
```

Setelah mesin melakukan proses pembagian data citra, mesin akan melakukan proses pre-processing. Pada tahap proses pre-processing kali ini, mesin akan menambahkan filter GaussianNoise senilai 0,2 yang akan digunakan untuk menghasilkan noise. Selain itu menambahkan filter Gaussian Noise, juga menambahkan Random Flip secara horizontal, Random Rotation, dan Random Zoom.

```
preprocess = tf.keras.Sequential([


tf.keras.layers.experimental.preprocessing.RandomFlip('horizontal'),
    tf.keras.layers.experimental.preprocessing.RandomRotation(0.2),
```

```
    tf.keras.layers.experimental.preprocessing.RandomZoom(0.2),
    tf.keras.layers.GaussianNoise(0.2)
])
```

Setelah mesin melakukan proses pre-processing, mesin akan menambahkan *ResNet* didalam objek base_model. Pada ResNet kali ini akan menerima inputan citra dengan ukuran 224 x 224 piksel dan 3 saluran warna (RGB) yang diatur dalam input_shape.

```
base_model = ResNet50(include_top=False, weights='imagenet',
input_shape=(224,224,3))
```

Setelah mesin melakukan penambahan layer untuk ResNet, mesin akan menambahkan racangan layer seperti dengan potongan code tersebut.

```
       model = tf.keras.Sequential([
           #augmentation,
           base_model,
           tf.keras.layers.Dropout(0.5),
           tf.keras.layers.Dense(128, activation='relu'),
           tf.keras.layers.Dropout(0.5),
           tf.keras.layers.Dense(512, activation='relu'),
           tf.keras.layers.Dense(len(train_generator.class_indices),
       activation='softmax')
])
```

Setelah mesin melakukan penambahan layer sesuai dengan rancangan, tahap selanjutnya adalah penambahan *optimizer*. Pada skenario uji coba kali ini dilakukan dengan membandingkan model dengan menggunakan *optimizer* Adam dan RMSprop dengan *learning rate* sebesar 0.0001.

1. *Optimizer* Adam dengan *learning rate* sebesar 0.0001

```
model.compile(optimizer=Adam(learning_rate=0.0001),
loss='categorical_crossentropy', metrics=['categorical_accuracy'])
```

2. *Optimizer* RMSProp dengan *learning rate* sebesar 0.0001

```
model.compile(optimizer=RMSprop(learning_rate=0.0001),
loss='categorical_crossentropy', metrics=['categorical_accuracy'])
```

Setelah mesin melakukan penambahan *optimizer*, maka tahap selanjutnya adalah tahap pelatihan (*training*). Proses training data ini akan melibatkan seluruh proses yang telah dirancang. Pada proses training data ini akan dilakukan sebanyak 50 epoch.

```
history = model.fit(train_generator,
                    validation_data=validation_generator,
                    batch_size=64,
                    epochs=50,
                   verbose=1)
```

Setelah mesin melakukan proses training, mesin akan mencetak hasil dari proses training data. Hasil dari proses training meliputi: grafik training, akurasi, precision, recall dan f1-score

1. Hasil proses training menggunakan grafik training

```
import matplotlib.pyplot as plt

metrics = history.history

plt.plot(history.epoch,             metrics['categorical_accuracy'],
metrics['val_categorical_accuracy'])

plt.legend(['categorical_accuracy', 'val_categorical_accuracy'])

plt.show()



plt.plot(history.epoch, metrics['loss'], metrics['val_loss'])

plt.legend(['loss', 'val_loss'])

plt.show()
```

2. Hasil proses training menggunakan akurasi, precision, recall dan f1-score

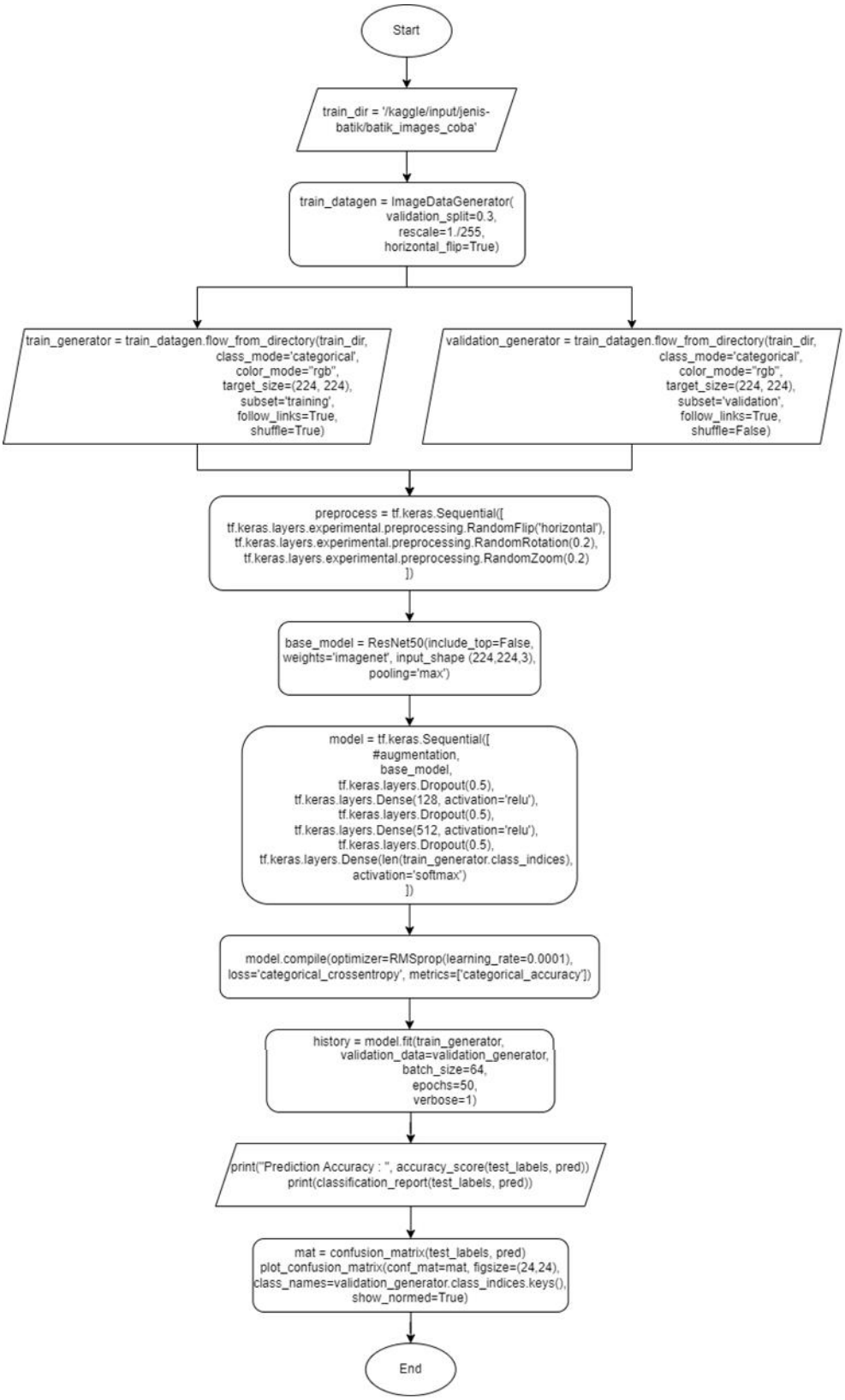
```
test_labels = validation_generator.classes

predictions = model.predict(validation_generator)

pred = np.argmax(predictions, axis=1)



from sklearn.metrics import accuracy_score

print("Prediction Accuracy : ", accuracy_score(test_labels, pred))



from sklearn.metrics import classification_report

print(classification_report(test_labels, pred))
```

Proses selanjutnya dari proses training citra batik adalah tahap pengujian. Uji coba pelatihan kali ini akan menggunakan metode *confusion matrix*.

```
from mlxtend.plotting import plot_confusion_matrix

from sklearn.metrics import confusion_matrix



mat = confusion_matrix(test_labels, pred)

plot_confusion_matrix(conf_mat=mat,              figsize=(24,24),
class_names=validation_generator.class_indices.keys(),
show normed=True)
```

5.1.4. Skenario 4

Uji coba ini dilakukan dengan membandingkan model dengan menggunakan *ResNet 50, Resnet 101, dan ResNet 152* dengan *optimizer* RMSprop dengan *learning rate* sebesar 0.0001 dan melakukan pelatihan sebanyak 50 epoch.



Gambar 5. 1 Flowchart Code Skenario 4

Pada skenario ini, yang membedakan dengan skenario yang lainnya adalah penggunaan layer *ResNet*.

1. Layer *ResNet 50*

```
base_model = ResNet50(include_top=False, weights='imagenet',
input_shape=(224,224,3))
```

2. Layer *ResNet 101*

```
base_model = ResNet101(include_top=False, weights='imagenet',
input_shape=(224,224,3))
```

3. Layer *ResNet 152*

```
base_model = ResNet152(include_top=False, weights='imagenet',
input_shape=(224,224,3))
```

BAB VI. HASIL DAN PEMBAHASAN

6.1. Hasil Implementasi Uji Coba

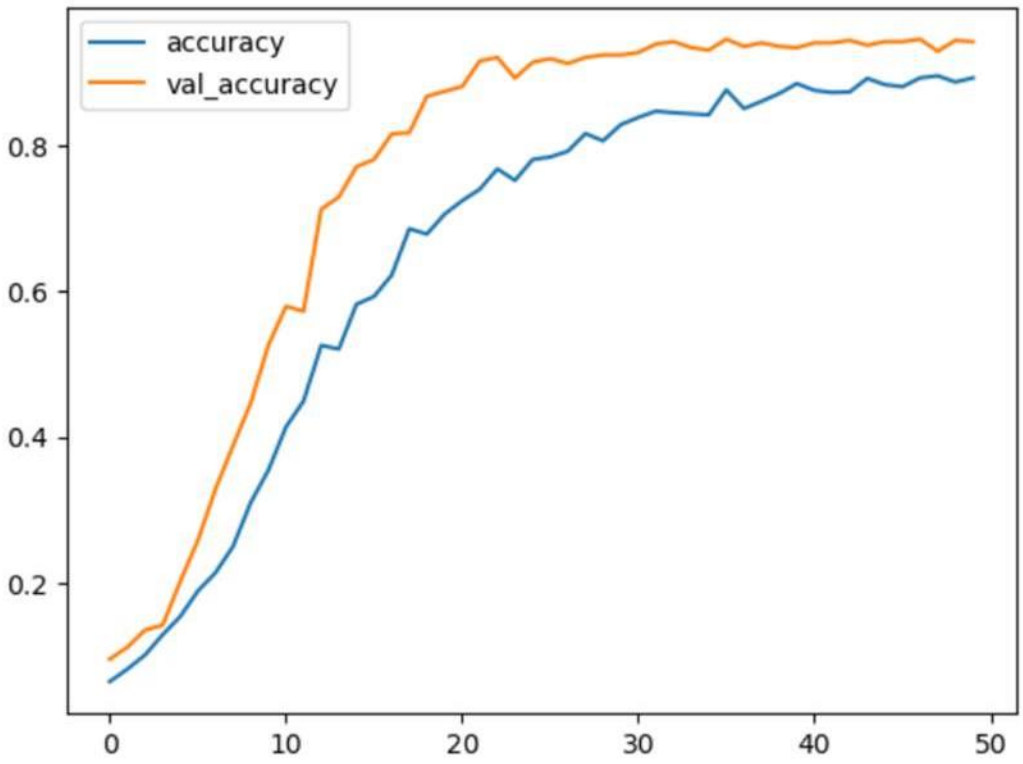
Hasil mplementasi Uji Coba *Image Classification* pada Jenis-Jenis Batik Menggunakan Algoritma *Convolutional Neural Network* dengan Model Arsitektur *ResNet* dengan beberapa skenario yang sudah dirancang untuk menemukan model pelatihan terbaik.

6.1.1. Hasil Uji Coba Skenario 1

Pada uji coba skenario 1 ini dilakukan dengan melakukan proses *training* dengan menggunakan algoritma *Convolutional Neural Network* tanpa menggunakan *ResNet*. Berikut merupakan hasil uji coba skenario 1.

5.1.1.1 Grafik *Training Accuracy*

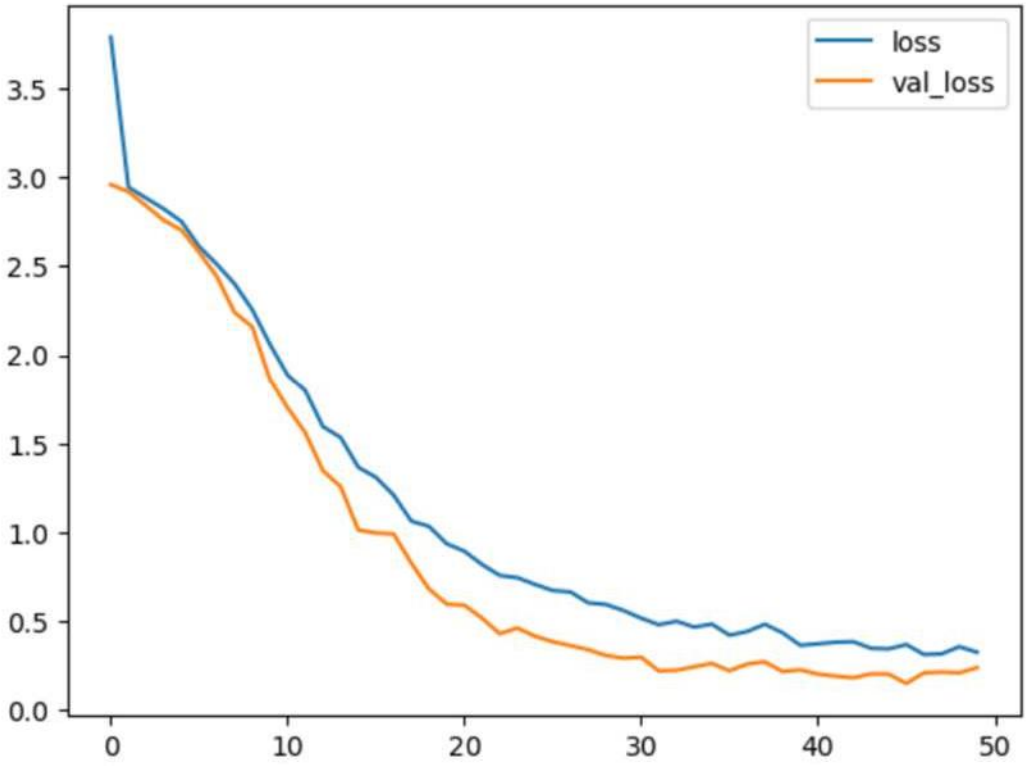
Berdasarkan proses *training* akan mendapatkan hasil berupa grafik *training* sebagai berikut.



Gambar 22 Hasil Grafik *Training Accuracy* Skenario 1

Berdasarkan grafik proses *training* pada gambar 6.1 menggunakan algoritma *Convolutional Neural Network* tanpa menggunakan *ResNet* mengalami kenaikan *accuracy* pada setiap epoch-nya. Hasil akhir pada proses *training* ini berada di atas 80% yang berakhir pada epoch ke 50.

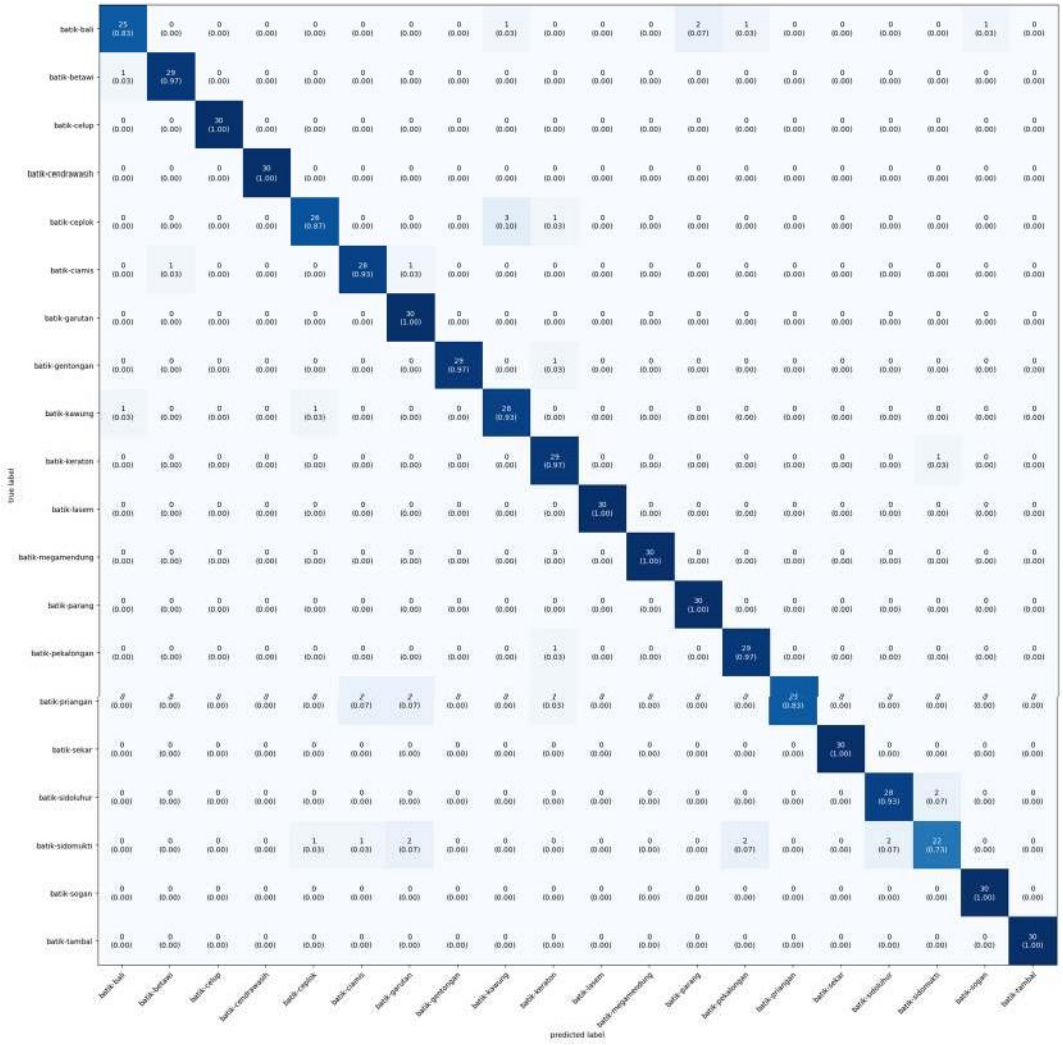
5.1.1.2 Grafik *Training loss*



Gambar 23 Hasil Grafik *Training Loss* Skenario 1

Berdasarkan grafik proses *training* pada gambar 6.2 menggunakan algoritma *Convolutional Neural Network* tanpa menggunakan *ResNet* mengalami penurunan *loss* pada setiap epoch-nya. Hasil akhir pada proses *training* ini berada di angka mendekati 0 yang berakhir pada epoch ke 50.

5.1.1.3 *Confusion Matrix*



Gambar 24 *Confusion Matrix* Skenario 1

Berdasarkan *confusion matrix* pada gambar 6.3 dapat dilihat bahwa proses *training* menggunakan algoritma *Convolutional Neural Network* tanpa menggunakan *ResNet* berhasil mengklasifikasikan dengan benar sebanyak 568 gambar. Berdasarkan hasil tersebut didapat hasil Batik Sidomukti yang mendapatkan jumlah terendah dalam pengklasifikasian menggunakan model ini. Batik Sidomukti hanya mendapatkan 22 gambar dari 30 gambar.

Pengujian ini dilakukan untuk mengukur presisi, sensitifitas, dan keakuratan dari data uji dan data latih terhadap hasil pengujian. Pengujian dilakukan dengan membandingkan antara hasil prediksi dengan hasil klasifikasi. Dari hasil *confusion matrix* ini bisa kita klasifikasikan sebagai berikut :

Tabel 7 Tabel Klasifikasi Data Skenario 1

Kelas	Klasifikasi				Jumlah
	TP	FN	FP	TN	
Batik Bali	25	5	2	568	600
Batik Betawi	29	1	1	569	600
Batik Celup	30	0	0	570	600
Batik Cendrawasih	30	0	0	570	600
Batik Ceplok	26	4	2	568	600
Batik Ciamis	28	2	3	567	600
Batik Garutan	30	0	5	565	600
Batik Gentongan	29	1	0	570	600
Batik Kawung	28	2	4	566	600
Batik Keraton	29	1	4	566	600
Batik Lasem	30	0	0	570	600
Batik Megamendung	30	0	0	570	600
Batik Parang	30	0	2	568	600
Batik Pekalongan	29	1	3	567	600
Batik Priangan	25	5	0	570	600
Batik Sekar	30	0	0	570	600
Batik Sidoluhur	28	2	2	568	600
Batik Sidomukti	22	8	3	567	600
Batik Sogan	30	0	1	569	600
Batik Tambal	30	0	0	570	600

Berdasarkan data klasifikasi pada Tabel 6.1 Klasifikasi Data dapat dihitung nilai presisi, sensitifitas, dan akurasinya sebagai berikut :

Tabel 8 Tabel Hasil Nilai dari Klasifikasi Data Skenario 1

Kelas	**Hasil**			
	Accuracy	***Precision***	***Recall***	***F1-Score***
Batik Bali	0,988333	0,833333	0,925926	0,877193
Batik Betawi	0,996667	0,966667	0,966667	0,966667
Batik Celup	1	1	1	1
Batik Cendrawasih	1	1	1	1
Batik Ceplok	0,99	0,866667	0,928571	0,896552
Batik Ciamis	0,991667	0,933333	0,903226	0,918033
Batik Garutan	0,991667	1	0,857143	0,923077
Batik Gentongan	0,998333	0,966667	1	0,983051
Batik Kawung	0,99	0,933333	0,875	0,903226
Batik Keraton	0,991667	0,966667	0,878788	0,920635
Batik Lasem	1	1	1	1
Batik Megamendung	1	1	1	1
Batik Parang	0,996667	1	0,9375	0,967742
Batik Pekalongan	0,993333	0,966667	0,90625	0,935484
Batik Priangan	0,991667	0,833333	1	0,909091
Batik Sekar	1	1	1	1
Batik Sidoluhur	0,993333	0,933333	0,933333	0,933333
Batik Sidomukti	0,981667	0,733333	0,88	0,8
Batik Sogan	0,998333	1	0,967742	0,983607
Batik Tambal	1	1	1	1

```
19/19 [==============================] - 8s 387ms/step
Prediction Accuracy :  0.9466666666666667
              precision    recall  f1-score   support

           0       0.93      0.83      0.88        30
           1       0.97      0.97      0.97        30
           2       1.00      1.00      1.00        30
           3       1.00      1.00      1.00        30
           4       0.93      0.87      0.90        30
           5       0.90      0.93      0.92        30
           6       0.86      1.00      0.92        30
           7       1.00      0.97      0.98        30
           8       0.88      0.93      0.90        30
           9       0.88      0.97      0.92        30
          10       1.00      1.00      1.00        30
          11       1.00      1.00      1.00        30
          12       0.94      1.00      0.97        30
          13       0.91      0.97      0.94        30
          14       1.00      0.83      0.91        30
          15       1.00      1.00      1.00        30
          16       0.93      0.93      0.93        30
          17       0.88      0.73      0.80        30
          18       0.97      1.00      0.98        30
          19       1.00      1.00      1.00        30

    accuracy                           0.95       600
   macro avg       0.95      0.95      0.95       600
weighted avg       0.95      0.95      0.95       600
```

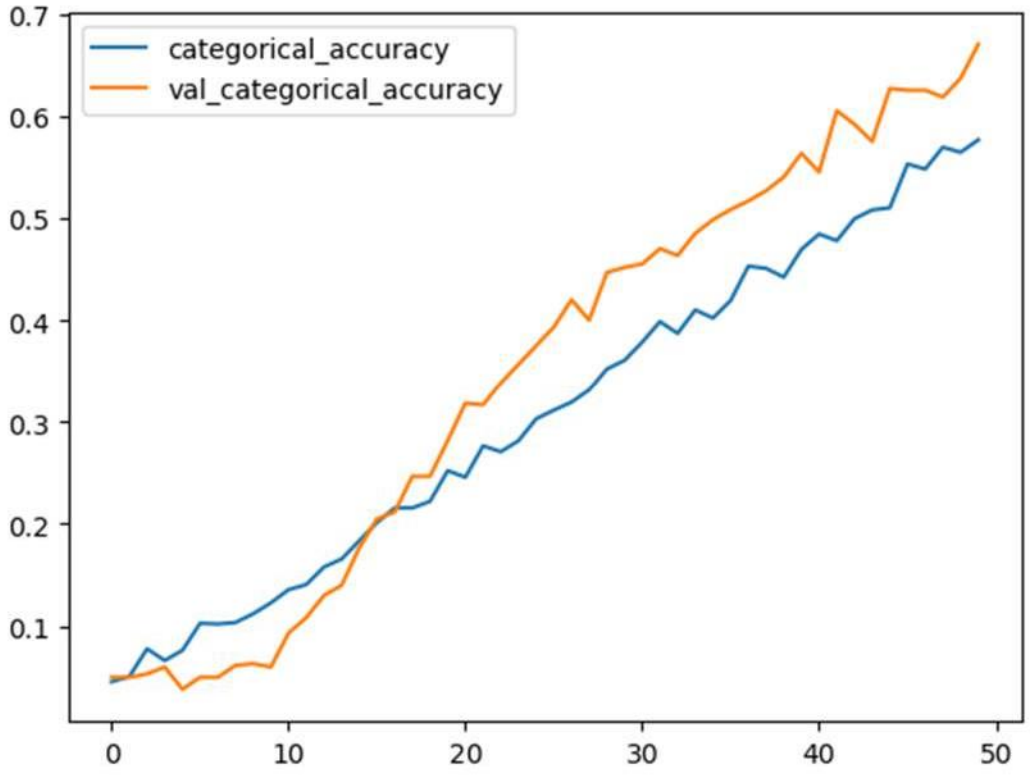
Gambar 25 Hasil *Accuracy* Skenario 1

Berdasarkan hasil *accuracy* pada gambar 6.4 dapat dilihat bahwa proses *training* menggunakan algoritma *Convolutional Neural Network* tanpa menggunakan *ResNet* memperoleh skor sebesar 95%.

6.1.2. Hasil Uji Coba Skenario 2

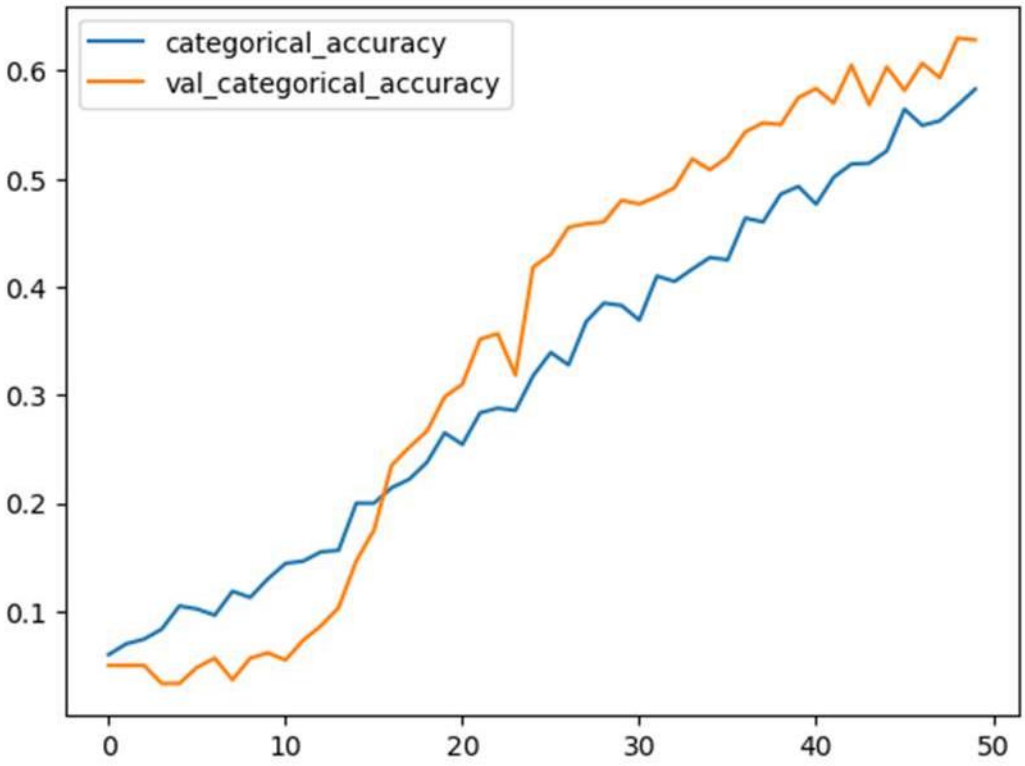
Pada uji coba skenario 2 ini dilakukan dengan membandingkan model dengan menggunakan data *preprocessing* dan dengan model tanpa *preprocessing. Preprocessing* yang digunakan pada uji coba ini yaitu menggunakan filter GaussianNoise dan akan melakukan pelatihan sebanyak 50 *epoch*. Berikut merupakan hasil uji coba skenario 2.

5.1.1.4 Grafik *Training Accuracy*



Gambar 26 Hasil Grafik *Training Accuracy* dengan Data *Preprocessing*

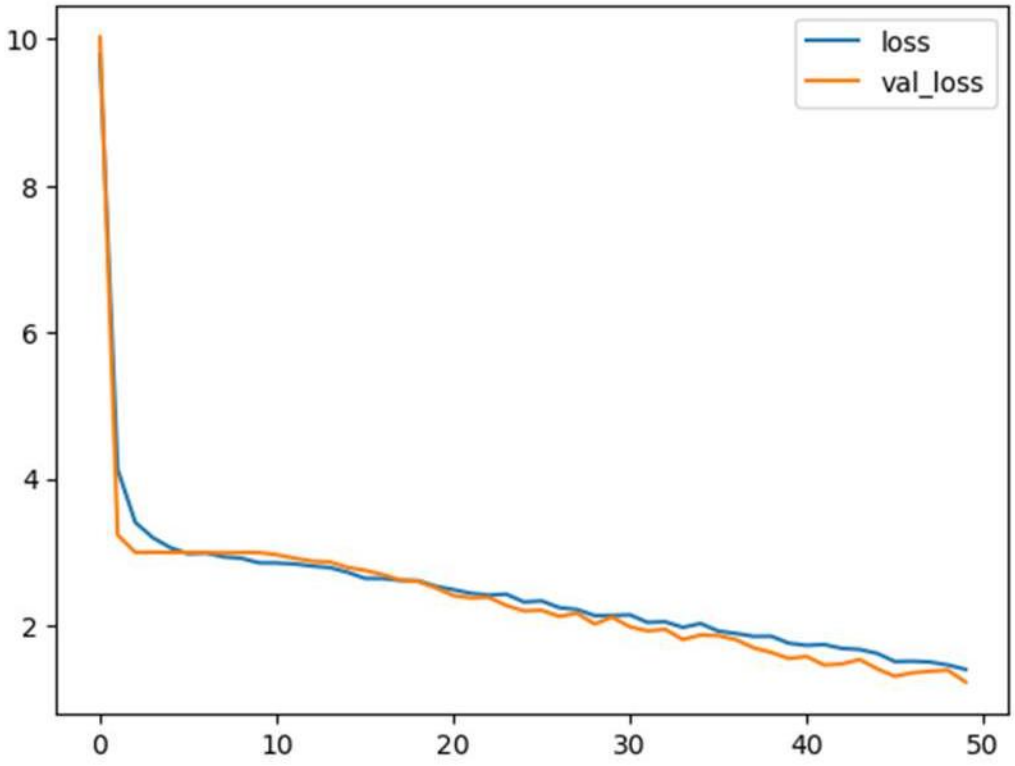
Berdasarkan grafik proses *training* pada gambar 6.5 dengan data *preprocessing* mengalami kenaikan *accuracy* pada setiap epoch-nya. Hasil akhir pada proses *training* ini hampir berada di 70% yang berakhir pada epoch ke 50.



Gambar 27 Grafik *Training Accuracy* tanpa Data *Preprocessing*

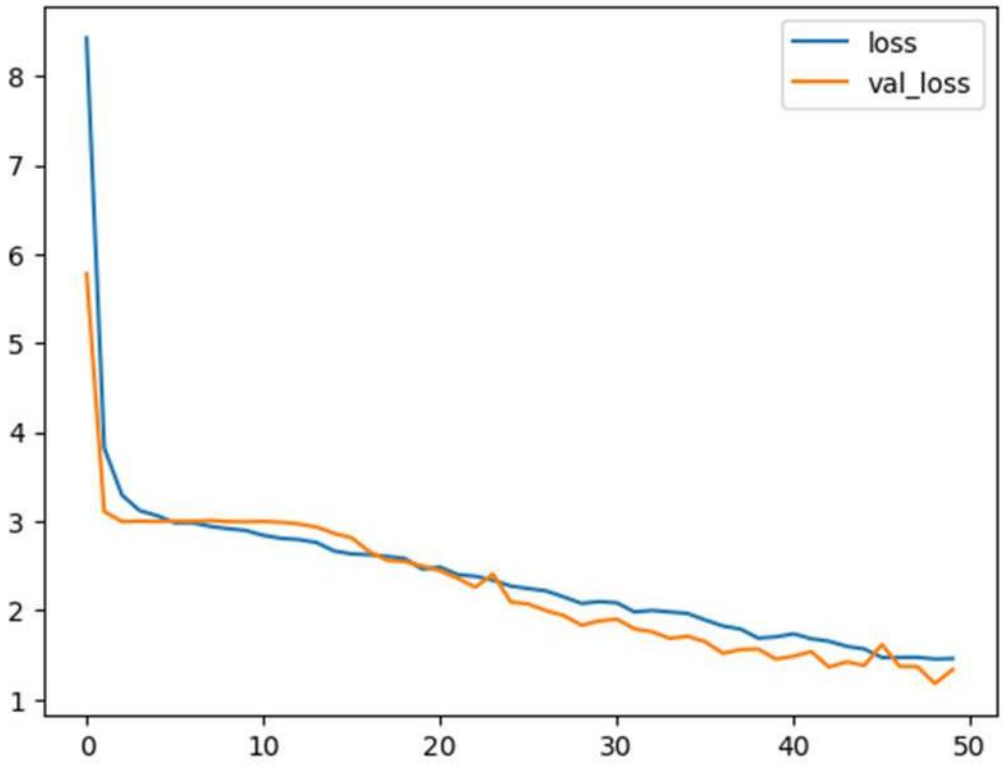
Berdasarkan grafik proses *training* pada gambar 6.6 tanpa data *preprocessing* mengalami kenaikan *accuracy* pada setiap epoch-nya. Hasil akhir pada proses *training* ini berada di atas 60% yang berakhir pada epoch ke 50.

5.1.1.5 Grafik *Training Loss*



Gambar 28 Hasil Grafik *Training Loss* dengan Data *Preprocessing*

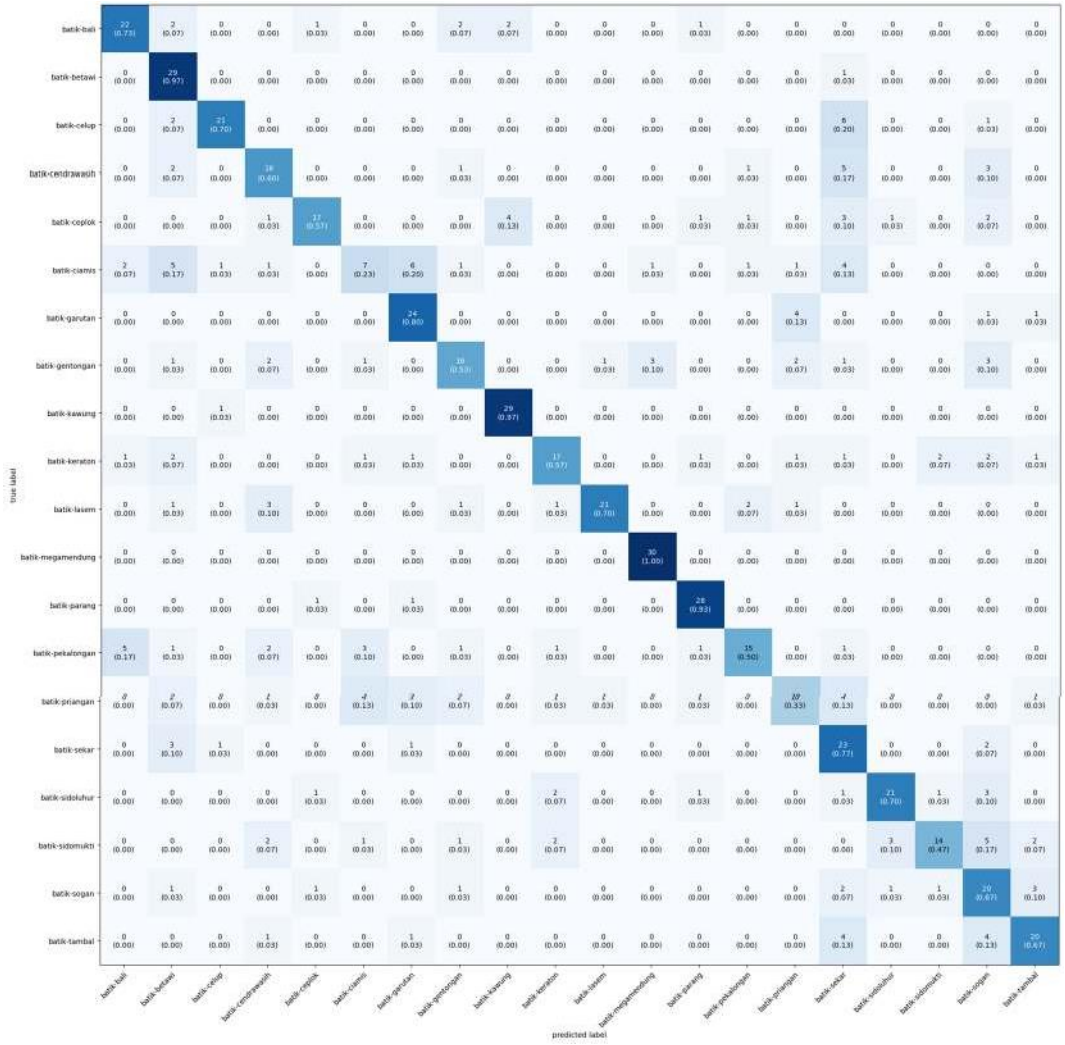
Berdasarkan grafik proses *training* pada gambar 6.7 dengan data *preprocessing* mengalami penurunan *loss* pada setiap epoch-nya. Hasil akhir pada proses *training* ini berada di angka mendekati 0 yang berakhir pada epoch ke 50.



Gambar 29 Hasil Grafik *Training Loss* Tanpa Data *Preprocessing*

Berdasarkan grafik proses *training* pada gambar 6.8 tanpa data *preprocessing* mengalami penurunan *loss* pada setiap epoch-nya. Hasil akhir pada proses *training* ini berada di angka mendekati 1 yang berakhir pada epoch ke 50.

5.1.1.6 *Confusion Matrix*



Gambar 30 *Confusion Matrix* Dengan Data *Preprocessing*

Berdasarkan *confusion matrix* pada gambar 6.9 dapat dilihat bahwa proses *training* tanpa menggunakan data *preprocessing* berhasil mengklasifikasikan dengan benar hanya sebanyak 402 gambar. Berdasarkan hasil tersebut didapat hasil Batik Ciamis yang mendapatkan jumlah terendah dalam pengklasifikasian menggunakan model ini. Batik Ciamis hanya mendapatkan 7 gambar dari 30 gambar.

Pengujian ini dilakukan untuk mengukur presisi, sensitifitas, dan keakuratan dari data uji dan data latih terhadap hasil pengujian. Pengujian dilakukan dengan membandingkan antara hasil prediksi dengan hasil klasifikasi. Dari hasil *confusion matrix* ini bisa kita klasifikasikan sebagai berikut :

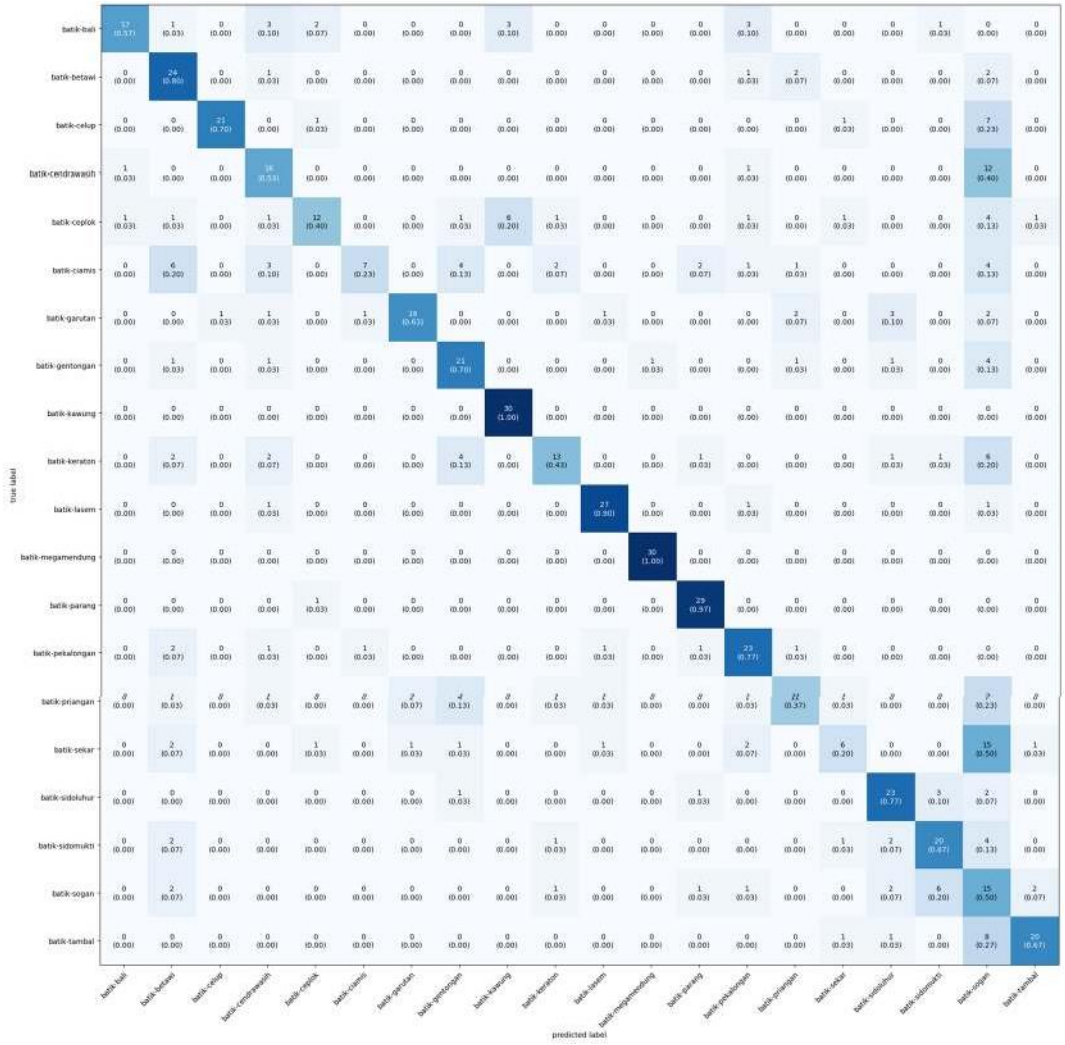
Tabel 9 Tabel Hasil Nilai dari Klasifikasi Data dengan *Preprocessing*

Kelas	**Klasifikasi**				**Jumlah**
	TP	**FP**	**FN**	**TN**	
Batik Bali	22	8	8	562	600
Batik Betawi	29	1	22	548	600
Batik Celup	21	3	9	567	600
Batik Cendrawasih	18	12	13	557	600
Batik Ceplok	17	13	4	566	600
Batik Ciamis	7	23	10	560	600
Batik Garutan	24	6	13	557	600
Batik Gentongan	16	14	10	560	600
Batik Kawung	29	1	6	564	600
Batik Keraton	17	13	7	563	600
Batik Lasem	21	9	2	568	600
Batik Megamendung	30	0	4	566	600
Batik Parang	28	2	6	564	600
Batik Pekalongan	15	15	5	565	600
Batik Priangan	10	20	9	561	600
Batik Sekar	23	7	33	537	600
Batik Sidoluhur	21	9	5	565	600
Batik Sidomukti	14	16	4	566	600
Batik Sogan	20	10	26	544	600
Batik Tambal	20	10	8	562	600

Berdasarkan data klasifikasi pada Tabel 6.3 Klasifikasi Data dapat dihitung nilai presisi, sensitifitas, dan akurasinya sebagai berikut :

Tabel 10 Tabel Hasil Nilai dari Klasifikasi Data dengan *Preprocessing*

Kelas	**Hasil**			
	Accuracy	***Recall***	***Precision***	***F1-Score***
Batik Bali	0,973333	0,733333	0,733333	0,733333
Batik Betawi	0,961667	0,966667	0,568627	0,716049
Batik Celup	0,98	0,875	0,7	0,777778
Batik Cendrawasih	0,958333	0,6	0,580645	0,590164
Batik Ceplok	0,971667	0,566667	0,809524	0,666667
Batik Ciamis	0,945	0,233333	0,411765	0,297872
Batik Garutan	0,968333	0,8	0,648649	0,716418
Batik Gentongan	0,96	0,533333	0,615385	0,571429
Batik Kawung	0,988333	0,966667	0,828571	0,892308
Batik Keraton	0,966667	0,566667	0,708333	0,62963
Batik Lasem	0,981667	0,7	0,913043	0,792453
Batik Megamendung	0,993333	1	0,882353	0,9375
Batik Parang	0,986667	0,933333	0,823529	0,875
Batik Pekalongan	0,966667	0,5	0,75	0,6
Batik Priangan	0,951667	0,333333	0,526316	0,408163
Batik Sekar	0,933333	0,766667	0,410714	0,534884
Batik Sidoluhur	0,976667	0,7	0,807692	0,75
Batik Sidomukti	0,966667	0,466667	0,777778	0,583333
Batik Sogan	0,94	0,666667	0,434783	0,526316
Batik Tambal	0,97	0,666667	0,714286	0,689655



Gambar 31 *Confusion Matrix* tanpa Data *Preprocessing*

Berdasarkan *confusion matrix* pada gambar 6.10 dapat dilihat bahwa proses *training* tanpa menggunakan data *preprocessing* berhasil mengklasifikasikan dengan benar hanya sebanyak 384 gambar. Berdasarkan hasil tersebut didapat hasil Batik Sekar yang mendapatkan jumlah terendah dalam pengklasifikasian menggunakan model ini. Batik Sekar hanya mendapatkan 11 gambar dari 30 gambar.

Pengujian ini dilakukan untuk mengukur presisi, sensitifitas, dan keakuratan dari data uji dan data latih terhadap hasil pengujian. Pengujian dilakukan dengan membandingkan antara hasil prediksi dengan hasil klasifikasi. Dari hasil *confusion matrix* ini bisa kita klasifikasikan sebagai berikut :

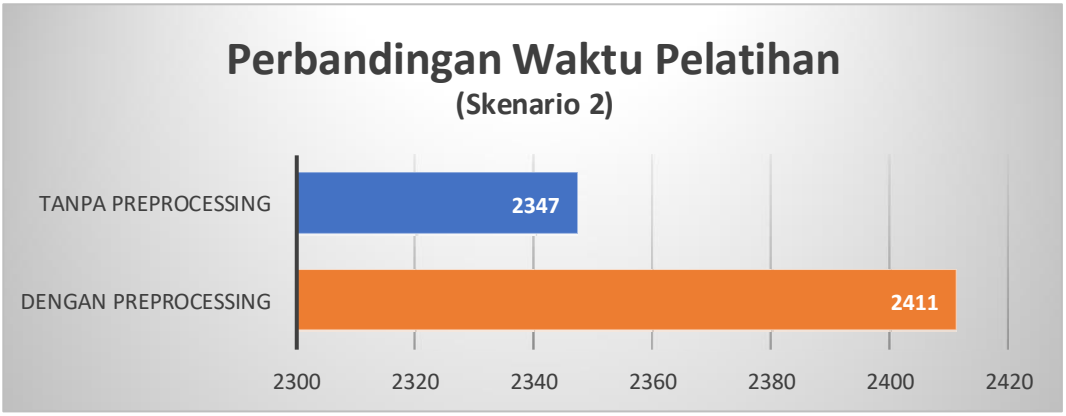
Tabel 11 Tabel Hasil Nilai dari Klasifikasi Data tanpa *Preprocessing*

Kelas	**Klasifikasi**				**Jumlah**
	TP	**FP**	**FN**	**TN**	
Batik Bali	17	13	2	568	600
Batik Betawi	24	6	20	550	600
Batik Celup	21	1	9	569	600
Batik Cendrawasih	16	14	15	555	600
Batik Ceplok	12	18	5	565	600
Batik Ciamis	7	23	2	568	600
Batik Garutan	19	11	3	567	600
Batik Gentongan	21	9	15	555	600
Batik Kawung	30	0	9	561	600
Batik Keraton	13	17	6	564	600
Batik Lasem	27	3	4	566	600
Batik Megamendung	30	0	1	569	600
Batik Parang	29	1	6	564	600
Batik Pekalongan	23	7	12	558	600
Batik Priangan	11	19	7	563	600
Batik Sekar	6	24	5	565	600
Batik Sidoluhur	23	7	10	560	600
Batik Sidomukti	20	10	11	559	600
Batik Sogan	15	15	78	492	600
Batik Tambal	20	10	4	566	600

Berdasarkan data klasifikasi pada Tabel 6.4 Klasifikasi Data dapat dihitung nilai presisi, sensitifitas, dan akurasinya sebagai berikut:

Tabel 12 Tabel Hasil Nilai dari Klasifikasi Data tanpa *Preprocessing*

Kelas	**Hasil**			
	Accuracy	***Recall***	***Precision***	***F1-Score***
Batik Bali	0,975	0,566667	0,894737	0,693878
Batik Betawi	0,956667	0,8	0,545455	0,648649
Batik Celup	0,983333	0,954545	0,7	0,807692
Batik Cendrawasih	0,951667	0,533333	0,516129	0,52459
Batik Ceplok	0,961667	0,4	0,705882	0,510638
Batik Ciamis	0,958333	0,233333	0,777778	0,358974
Batik Garutan	0,976667	0,633333	0,863636	0,730769
Batik Gentongan	0,96	0,7	0,583333	0,636364
Batik Kawung	0,985	1	0,769231	0,869565
Batik Keraton	0,961667	0,433333	0,684211	0,530612
Batik Lasem	0,988333	0,9	0,870968	0,885246
Batik Megamendung	0,998333	1	0,967742	0,983607
Batik Parang	0,988333	0,966667	0,828571	0,892308
Batik Pekalongan	0,968333	0,766667	0,657143	0,707692
Batik Priangan	0,956667	0,366667	0,611111	0,458333
Batik Sekar	0,951667	0,2	0,545455	0,292683
Batik Sidoluhur	0,971667	0,766667	0,69697	0,730159
Batik Sidomukti	0,965	0,666667	0,645161	0,655738
Batik Sogan	0,845	0,5	0,16129	0,243902
Batik Tambal	0,976667	0,666667	0,833333	0,740741



Gambar 32 Perbandingan Waktu yang Digunakan untuk Pelatihan pada Skenario 2

Pada gambar 6.11 menunjukkan bahwa pelatihan dengan menggunakan data *preprocessing* mendapatkan waktu yang lebih cepat daripada tiddak menggunakan data pelatihan. Hal ini menunjukkan bahwa dengan memberikan data *preprocessing gaussian noise* juga dapat mempercepat mesin untuk melakukan pelatihan.

Tabel 13 *classification metrics* skenario 2

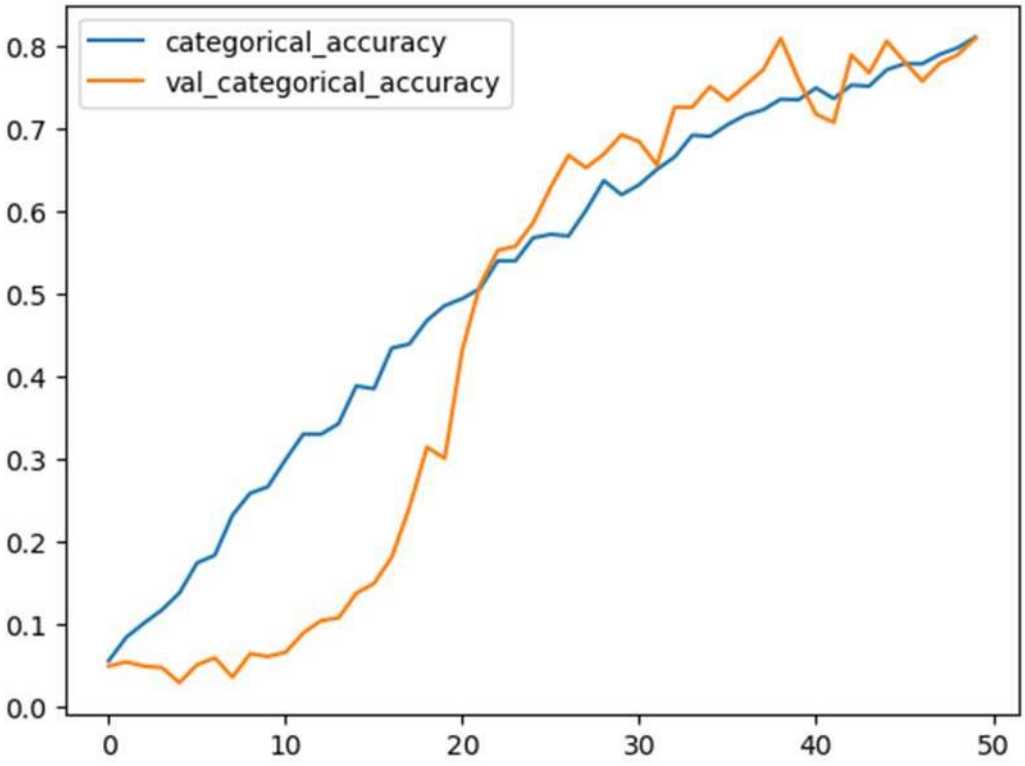
Skenario	*Accuracy* pada Sistem	*Accuracy* pada perhitungan manual	*Precision*	*Recall*	*F1-score*
Tanpa *pre-processing*	0.64	0.967	0.71	0.64	0.65
Dengan *pre-processing*	0.67	0.964	0.69	0.67	0.66

Tabel 6.6 menunjukkan bahwa kedua percobaan memiliki nilai *accuracy* yang sudah baik tapi untuk *precision*, *recall* dan *f1-score* masih mendapatkan hasil yang kurang baik dibawah 0.9. Namun dari kedelapan nilai dari percobaan pelatihan menggunakan data *preprocessing* tetap lebih tinggi dibandingkan dengan percobaan pelatihan tanpa menggunaka n data *preprocessing.*

6.1.3. Hasil Uji Coba Skenario 3

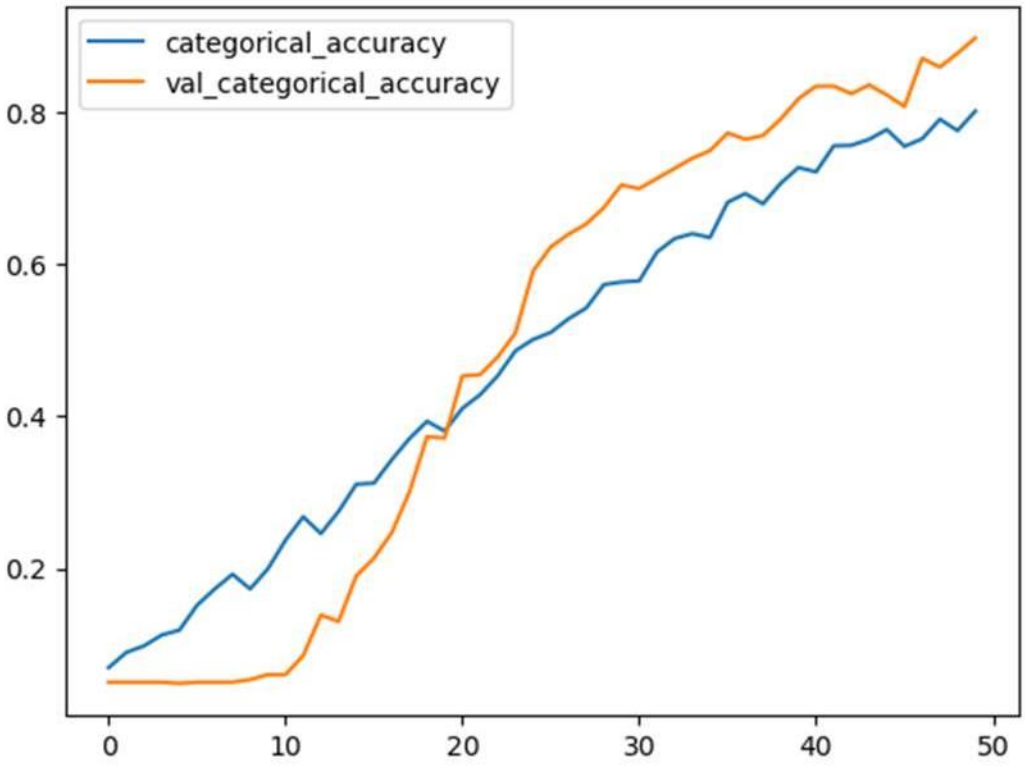
Pada uji coba skenario 3 ini dilakukan dengan membandingkan membandingkan model dengan menggunakan *optimizer* Adam dan RMSprop dengan *learning rate* sebesar 0.0001 dan melakukan pelatihan sebanyak 50 epoch. Berikut merupakan hasil uji coba skenario 3.

5.1.1.7 Grafik *Training Accuracy*



Gambar 33 Hasil Grafik *Training Accuracy* menggunakan *Optimizer* RMSprop dengan *learning rate* 0.0001

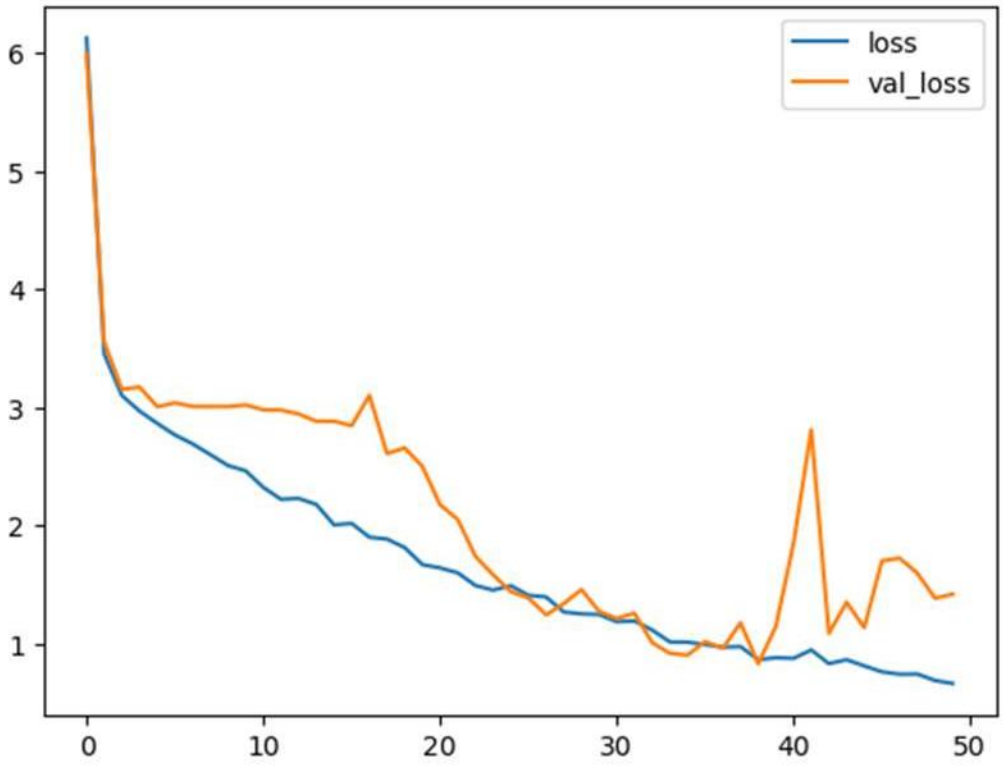
Berdasarkan grafik proses *training* pada gambar 6.11 menggunakan *optimizer* RMSprop dengan *learning rate* sebesar 0.0001 mengalami kenaikan *accuracy* pada setiap epoch-nya. Pada data validasi pada epoch 39 menuju epoch 40 sempat mengalami penurunan yang cukup lumayan dari 0.81 ke 0.76. Kemudian mengalami kenaikan lagi di epoch 42 menuju ke epoch 43 dari 0.70 ke 0.79. Hal ini terjadi berulang kali hingga ke epoch 50. Hasil akhir pada proses *training* ini berada di atas 80% yang berakhir pada epoch ke 50.



Gambar 34 Hasil Grafik *Training Accuracy* menggunakan *Optimizer* Adam dengan *learning rate* 0.0001

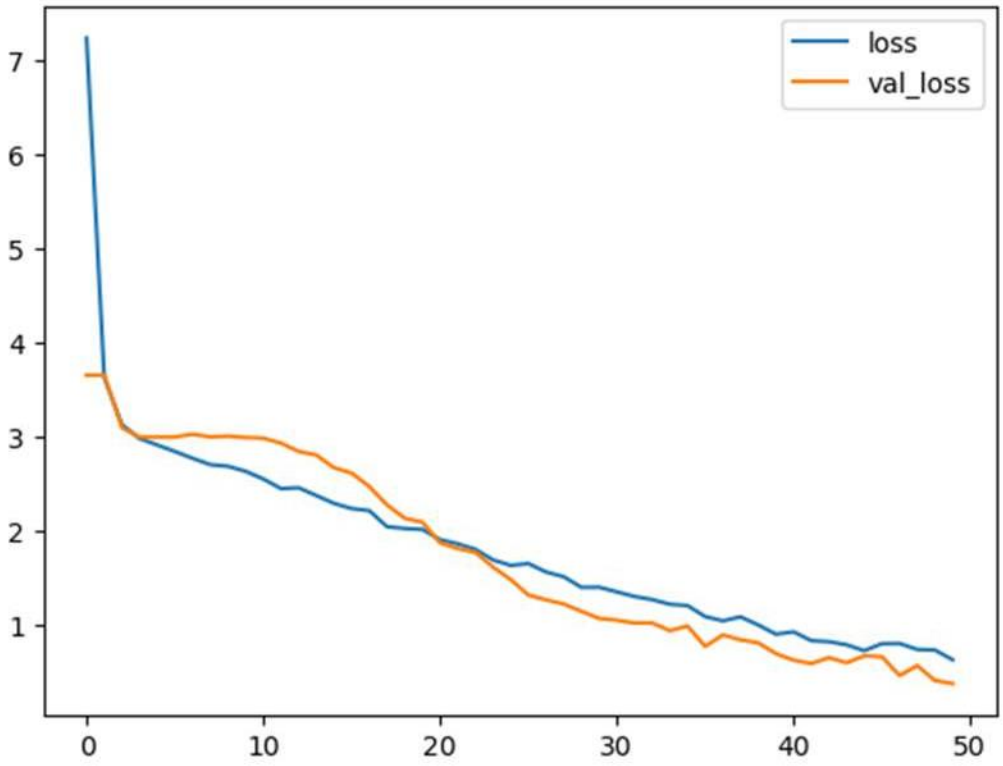
Berdasarkan grafik proses *training* pada gambar 6.12 menggunakan *optimizer* Adam dengan *learning rate* sebesar 0.0001 mengalami kenaikan *accuracy* pada setiap epoch-nya. Hasil akhir pada proses *training* ini berada di atas 80% yang berakhir pada epoch ke 50.

5.1.1.8 Grafik *Training Loss*



Gambar 35 Hasil Grafik *Training Loss* menggunakan *Optimizer* RMSprop dengan *learning rate* 0.0001

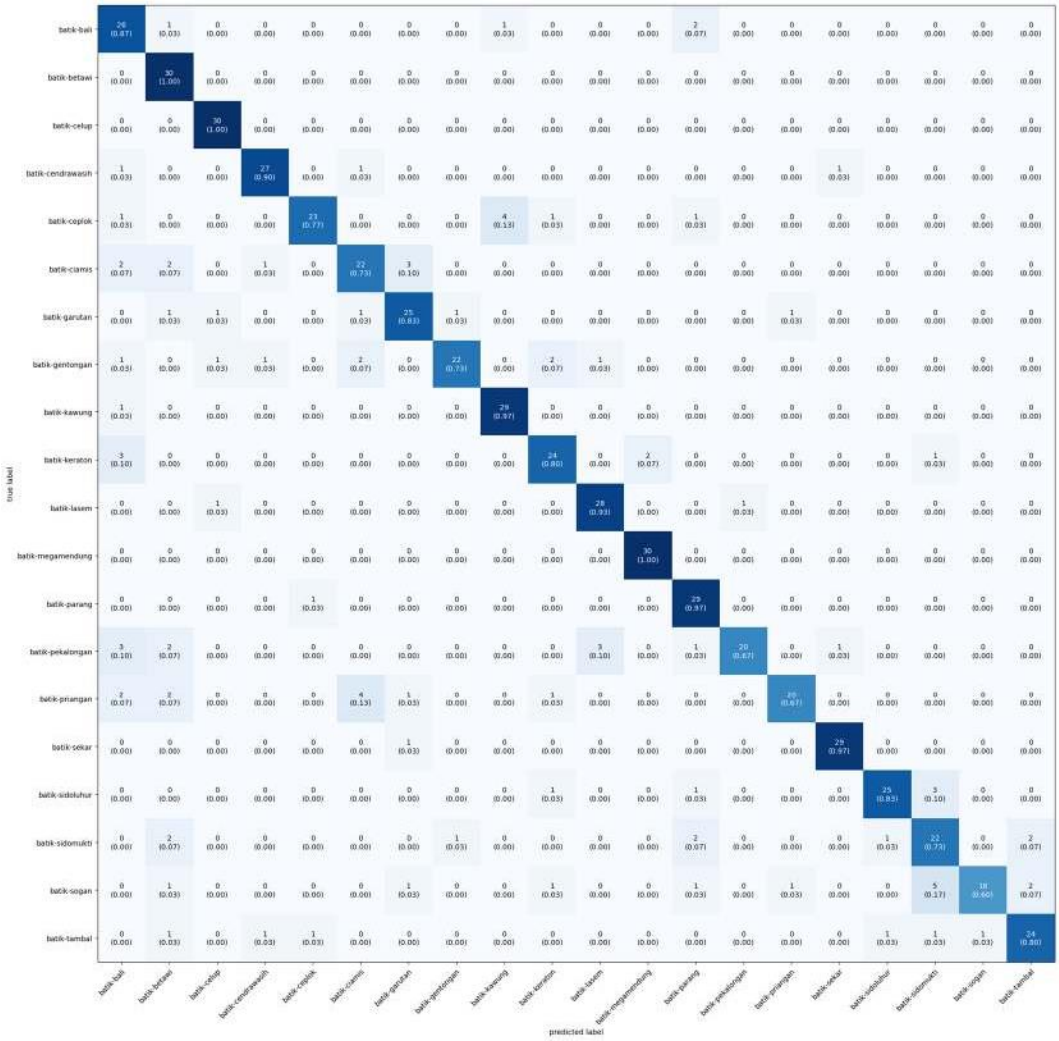
Berdasarkan grafik proses *training* pada gambar 6.13 menggunakan *Optimizer* RMSprop dengan *learning rate* 0.0001 mengalami penurunan *loss* yang tidak stabil setiap epoch-nya. Pada data validasi di epoch 41 menuju epoch 42 mengalami kenaikan yang cukup drastis dari 1.8 ke 2.8, namun langsung mengalami penurunan kembali pada epoch 42 menuju epoch 43 dari 2.8 ke 1.1. Hal ini terjadi berulang kali hingga ke epoch 50. Dan hasil akhir pada proses *training* ini berada di sekitar angka 1.3 yang berakhir pada epoch ke 50.



Gambar 36 Hasil Grafik *Training Loss* menggunakan *Optimizer* Adam dengan *learning rate* 0.0001

Berdasarkan grafik proses *training* pada gambar 6.14 menggunakan *Optimizer* Adam dengan *learning rate* 0.0001 mengalami penurunan *loss* pada setiap epoch-nya. Hasil akhir pada proses *training* ini berada di angka mendekati 0 yang berakhir pada epoch ke 50.

5.1.1.9 *Confusion Matrix*



Gambar 37 *Confusion Matrix* menggunakan *Optimizer* RMSprop dengan *learning rate* 0.0001

Berdasarkan *confusion matrix* pada gambar 6.15 dapat dilihat bahwa proses *training* menggunakan *Optimizer* RMSprop dengan *learning rate* 0.0001 berhasil mengklasifikasikan dengan benar sebanyak 503 gambar. Berdasarkan hasil tersebut didapat hasil Batik Sogan yang mendapatkan jumlah terendah dalam pengklasifikasian menggunakan model ini. Batik Sogan hanya mendapatkan 18 gambar dari 30 gambar.

Pengujian ini dilakukan untuk mengukur presisi, sensitifitas, dan keakuratan dari data uji dan data latih terhadap hasil pengujian. Pengujian dilakukan dengan membandingkan antara hasil prediksi dengan hasil klasifikasi. Dari hasil *confusion matrix* ini bisa kita klasifikasikan sebagai berikut :

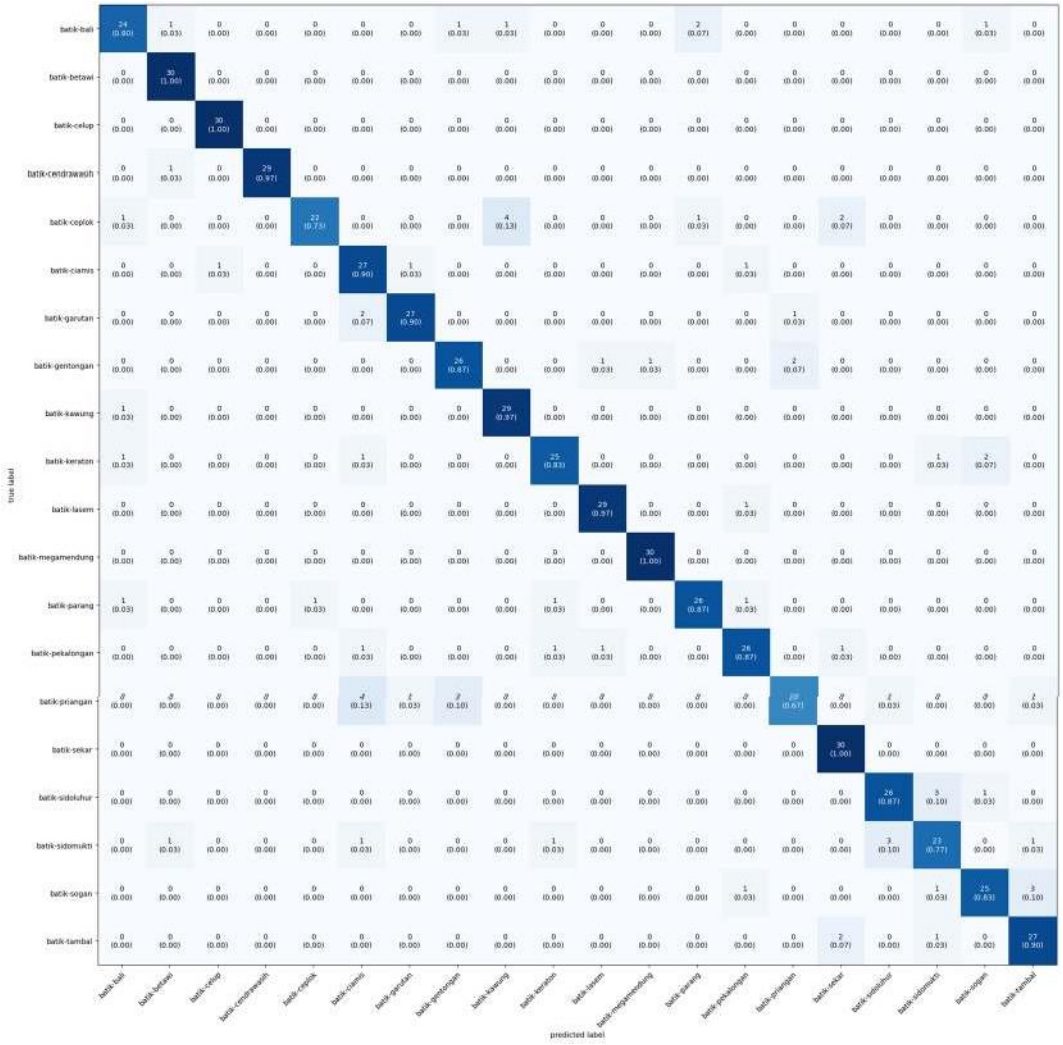
Tabel 14 Tabel Hasil Nilai dari Klasifikasi Data menggunakan *Optimizer* RMSprop dengan *learning rate* 0.0001

Kelas	**Klasifikasi**				**Jumlah**
	TP	**FP**	**FN**	**TN**	
Batik Bali	26	4	14	556	600
Batik Betawi	30	0	12	558	600
Batik Celup	30	3	0	567	600
Batik Cendrawasih	27	3	3	567	600
Batik Ceplok	23	7	2	568	600
Batik Ciamis	22	8	8	562	600
Batik Garutan	25	5	6	564	600
Batik Gentongan	22	8	2	568	600
Batik Kawung	29	1	5	565	600
Batik Keraton	24	6	6	564	600
Batik Lasem	28	2	4	566	600
Batik Megamendung	30	0	2	568	600
Batik Parang	29	1	8	562	600
Batik Pekalongan	20	10	1	569	600
Batik Priangan	20	10	2	568	600
Batik Sekar	29	1	2	568	600
Batik Sidoluhur	25	5	2	568	600
Batik Sidomukti	22	8	10	560	600
Batik Sogan	18	12	1	569	600
Batik Tambal	24	6	4	566	600

Berdasarkan data klasifikasi pada Tabel 6.8 Klasifikasi Data dapat dihitung nilai presisi, sensitifitas, dan akurasinya sebagai berikut:

Tabel 15 Hasil Nilai dari Klasifikasi Data menggunakan *Optimizer* RMSprop dengan *learning rate* 0.0001

Kelas	**Hasil**			
	Accuracy	***Recall***	***Precision***	***F1-Score***
Batik Bali	0,97	0,866667	0,65	0,742857
Batik Betawi	0,98	1	0,714286	0,833333
Batik Celup	0,995	0,909091	1	0,952381
Batik Cendrawasih	0,99	0,9	0,9	0,9
Batik Ceplok	0,985	0,766667	0,92	0,836364
Batik Ciamis	0,973333	0,733333	0,733333	0,733333
Batik Garutan	0,981667	0,833333	0,806452	0,819672
Batik Gentongan	0,983333	0,733333	0,916667	0,814815
Batik Kawung	0,99	0,966667	0,852941	0,90625
Batik Keraton	0,98	0,8	0,8	0,8
Batik Lasem	0,99	0,933333	0,875	0,903226
Batik Megamendung	0,996667	1	0,9375	0,967742
Batik Parang	0,985	0,966667	0,783784	0,865672
Batik Pekalongan	0,981667	0,666667	0,952381	0,784314
Batik Priangan	0,98	0,666667	0,909091	0,769231
Batik Sekar	0,995	0,966667	0,935484	0,95082
Batik Sidoluhur	0,988333	0,833333	0,925926	0,877193
Batik Sidomukti	0,97	0,733333	0,6875	0,709677
Batik Sogan	0,978333	0,6	0,947368	0,734694
Batik Tambal	0,983333	0,8	0,857143	0,827586



Gambar 38 *Confusion Matrix* menggunakan *Optimizer* Adam dengan *learning rate* 0.0001

Berdasarkan *confusion matrix* pada gambar 6.16 dapat dilihat bahwa proses *training* menggunakan *Optimizer* Adam dengan *learning rate* 0.0001 berhasil mengklasifikasikan dengan benar sebanyak 531 gambar. Berdasarkan hasil tersebut didapat hasil Batik Priangan yang mendapatkan jumlah terendah dalam pengklasifikasian menggunakan model ini. Batik Priangan hanya mendapatkan 20 gambar dari 30 gambar.

Pengujian ini dilakukan untuk mengukur presisi, sensitifitas, dan keakuratan dari data uji dan data latih terhadap hasil pengujian. Pengujian dilakukan dengan membandingkan antara hasil prediksi dengan hasil klasifikasi. Dari hasil *confusion matrix* ini bisa kita klasifikasikan sebagai berikut :

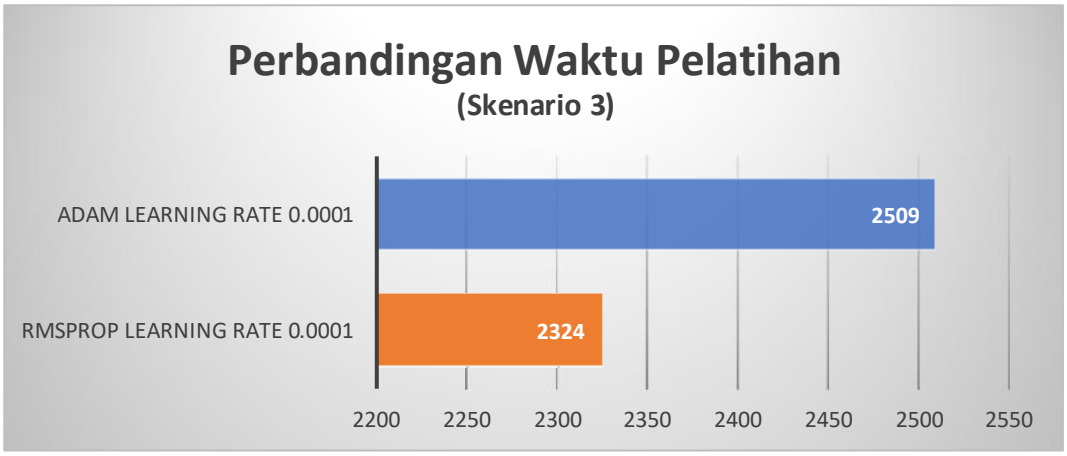
Tabel 16 Tabel Hasil Nilai dari Klasifikasi Data menggunakan *Optimizer* Adam dengan *learning rate* 0.0001

Kelas	**Klasifikasi**				**Jumlah**
	TP	**FN**	**FP**	**TN**	
Batik Bali	24	6	4	566	600
Batik Betawi	30	0	3	567	600
Batik Celup	30	1	0	569	600
Batik Cendrawasih	29	1	0	570	600
Batik Ceplok	22	8	1	569	600
Batik Ciamis	27	3	9	561	600
Batik Garutan	27	3	2	568	600
Batik Gentongan	26	4	4	566	600
Batik Kawung	29	1	5	565	600
Batik Keraton	25	5	3	567	600
Batik Lasem	29	1	2	568	600
Batik Megamendung	30	0	1	569	600
Batik Parang	26	4	3	567	600
Batik Pekalongan	26	4	4	566	600
Batik Priangan	20	10	3	567	600
Batik Sekar	30	0	5	565	600
Batik Sidoluhur	26	4	4	566	600
Batik Sidomukti	23	7	6	564	600
Batik Sogan	25	5	4	566	600
Batik Tambal	27	3	5	565	600

Berdasarkan data klasifikasi pada Tabel 6.9 Klasifikasi Data dapat dihitung nilai presisi, sensitifitas, dan akurasinya sebagai berikut:

Tabel 17 Hasil Nilai dari Klasifikasi Data menggunakan *Optimizer* Adam dengan *learning rate* 0.0001

Kelas	**Hasil**			
	Accuracy	***Recall***	***Precision***	***F1-Score***
Batik Bali	0,983333	0,8	0,857143	0,827586
Batik Betawi	0,995	1	0,909091	0,952381
Batik Celup	0,998333	0,967742	1	0,983607
Batik Cendrawasih	0,998333	0,966667	1	0,983051
Batik Ceplok	0,985	0,733333	0,956522	0,830189
Batik Ciamis	0,98	0,9	0,75	0,818182
Batik Garutan	0,991667	0,9	0,931034	0,915254
Batik Gentongan	0,986667	0,866667	0,866667	0,866667
Batik Kawung	0,99	0,966667	0,852941	0,90625
Batik Keraton	0,986667	0,833333	0,892857	0,862069
Batik Lasem	0,995	0,966667	0,935484	0,95082
Batik Megamendung	0,998333	1	0,967742	0,983607
Batik Parang	0,988333	0,866667	0,896552	0,881356
Batik Pekalongan	0,986667	0,866667	0,866667	0,866667
Batik Priangan	0,978333	0,666667	0,869565	0,754717
Batik Sekar	0,991667	1	0,857143	0,923077
Batik Sidoluhur	0,986667	0,866667	0,866667	0,866667
Batik Sidomukti	0,978333	0,766667	0,793103	0,779661
Batik Sogan	0,985	0,833333	0,862069	0,847458
Batik Tambal	0,986667	0,9	0,84375	0,870968



Gambar 39 Perbandingan Waktu yang Digunakan untuk Pelatihan pada Skenario 3

Pada gambar 6.18 menunjukkan bahwa pelatihan menggunakan *optimizer* RMSprop dengan *learning rate* 0.0001 mendapatkan waktu yang lebih cepat daripada menggunakan *optimizer* Adam dengan *learning rate* 0.0001.

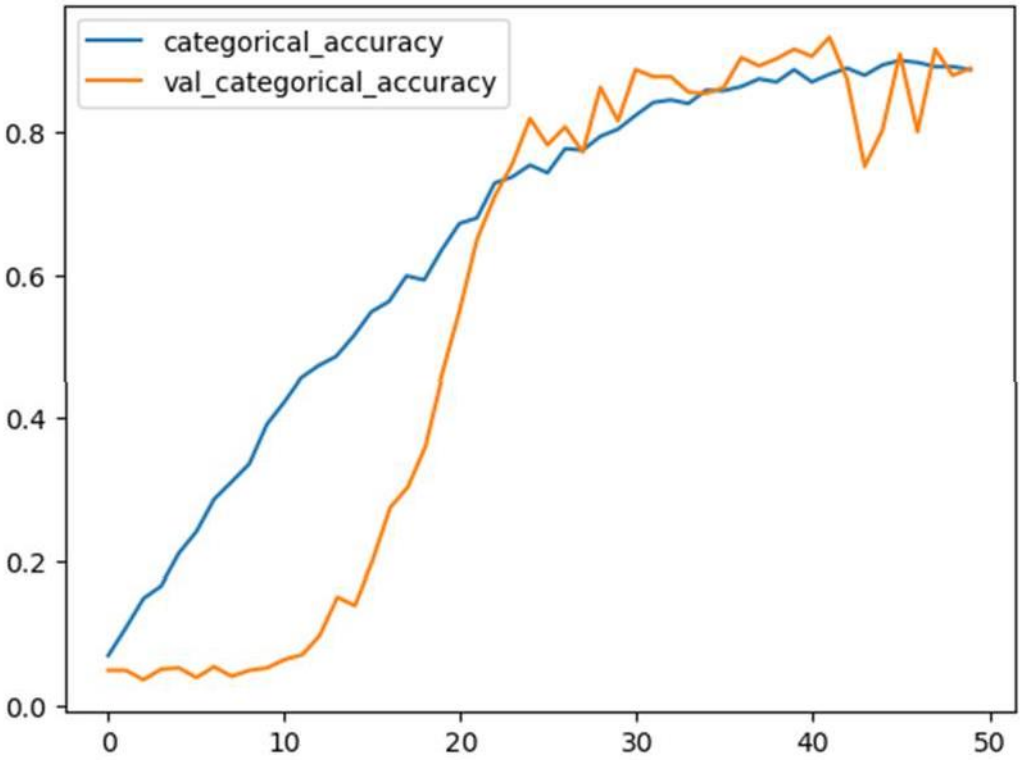
Tabel 6. 11 Tabel *classification metrics* skenario 3

Skenario	*Accuracy* pada Sistem	*Accuracy* pada perhitungan manual	*Precision*	*Recall*	*F1-score*
RMSprop (0.0001)	0.84	0,983833	0.85	0.84	0.84
Adam (0.0001)	0.89	0,9885	0.89	0.89	0.88

Berdasarkan uji coba membandingkan model antara mengguakan *optimizer* RMSprop dengan *learning rate* 0.0001 dan menggunakan *optimizer* Adam dengan *learning rate* 0.0001 menunjukkan bahwa kedua percobaan menggunakan *optimizer* RMSprop dengan *learning rate* 0.0001 memiliki nilai *accuracy* lebih tinggi daripada menggunakan *optimizer* Adam dengan *learning rate* 0.0001. Sedangkan pada nilai *precision* dari kedua uji coba telah menunjukkan angka yang hampir di angka 0.9, dimana hal ini menandakan bahwa model pelatihan sudah sangat baik dalam mengenali setiap kelas yang ada pada pelatihan meskipun terdapat perbedaan nilai antara *optimizer* RMSprop dengan *optimizer* Adam. Untuk nilai *recall* dan *f1-score* diatas 0.8, dimana ini merupakan hasil pelatihan yang baik. Perbandingan ini bisa dilihat pada tabel 6.11

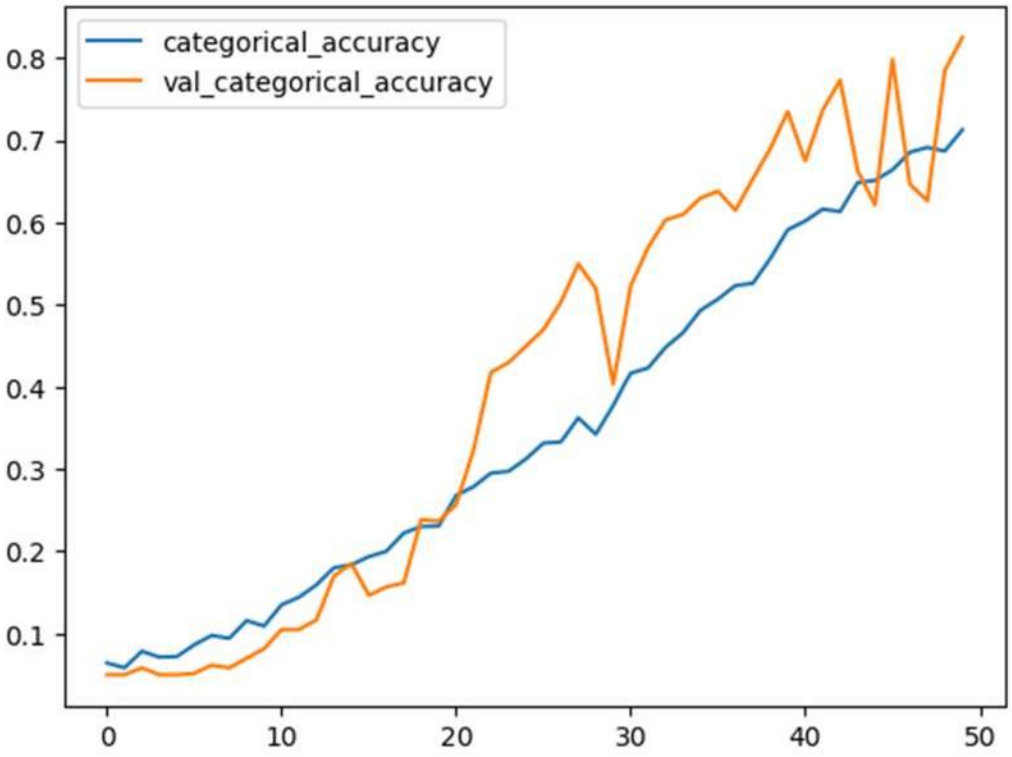
6.1.4. Hasil Uji Coba Skenario 4

Pada uji coba skenario 4 ini dilakukan dengan membandingkan model menggunakan *ResNet50, ResNet 101, dan ResNet152* dengan *optimizer* RMSprop dengan *learning rate* sebesar 0.0001 dan melakukan pelatihan sebanyak 50 epoch. Berdasarkan proses *training* akan mendapatkan hasil berupa grafik *training* sebagai berikut :



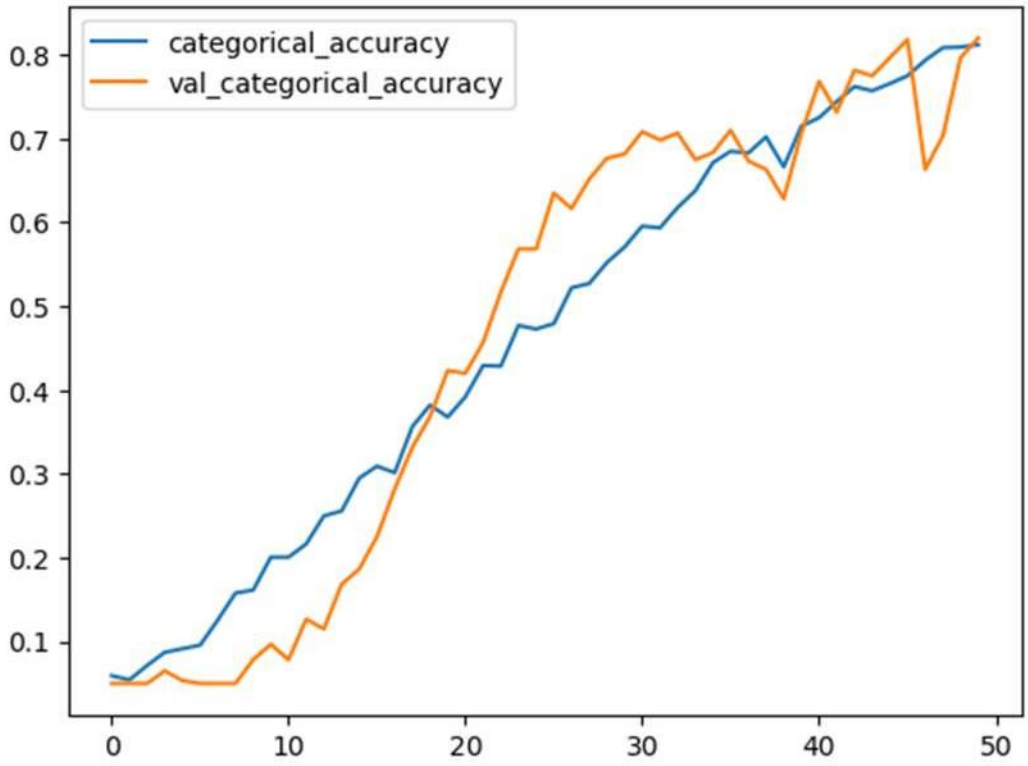
Gambar 40 Hasil Grafik *Training Accuracy* menggunakan *ResNet50*

Berdasarkan grafik proses *training* pada gambar 6.17 menggunakan *ResNet50* mengalami kenaikan *accuracy* yang tidak stabil pada setiap epoch-nya. Pada data validasi di epoch 43 menuju epoch 44 juga sempat mengalami penurunan terbesar dari 0.90 ke 0.73. Kemudian di epoch 47 menuju 48 mengalami kenaikan dari 0.80 ke 0.91. Begitu juga dengan data *training*, namun tidak terlalu naik turun seperti data validasi. Dan hasil akhir pada proses *training* ini berada di atas 80% yang berakhir pada epoch ke 50.



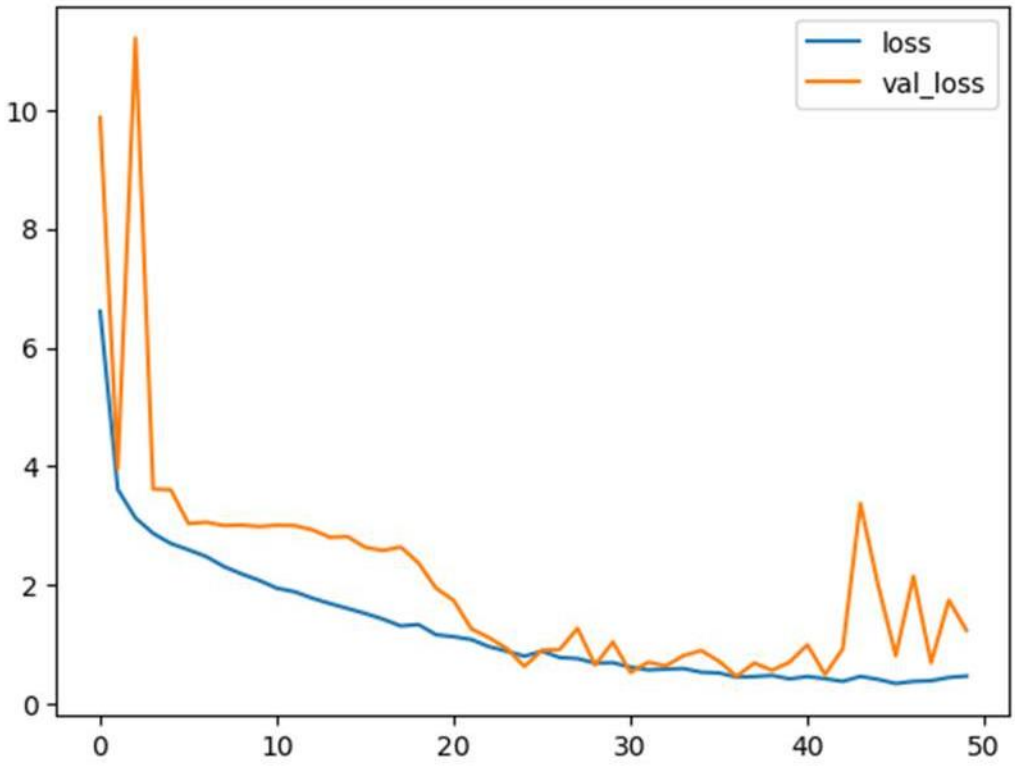
Gambar 41 Hasil Grafik *Training Accuracy* menggunakan *ResNet101*

Berdasarkan grafik proses *training* pada gambar 6.18 menggunakan *ResNet101* mengalami kenaikan *accuracy* yang tidak stabil pada setiap epoch-nya. Pada data validasi di epoch 29 menuju epoch 30 juga sempat mengalami penurunan drastis dari 0.52 ke 0.40. Namun di epoch 30 menuju epoch 31 langsung mengalami kenaikan dari 0.40 ke 0.53. Kemudian setelah mulai stabil mengalami penurunan lagi pada epoch 43 menuju epoch 44. Dan mengalami kenaikan lagi pada epoch 45 menuju epoch 46 dari 0.62 ke 0.78. Kemudian hal itu terjadi berulang kali, begitu juga dengan data *training*, namun tidak terlalu naik turun seperti data validasi. Dan hasil akhir pada proses *training* ini berada di atas 80% yang berakhir pada epoch ke 50.



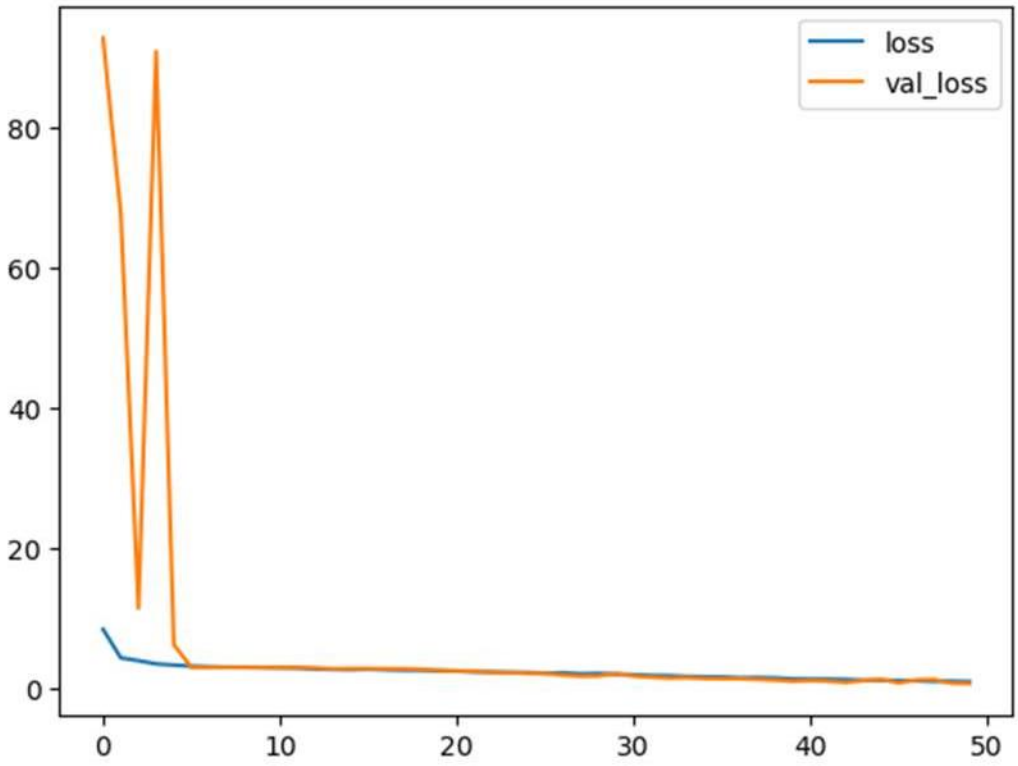
Gambar 42 Hasil Grafik *Training Accuracy* menggunakan *ResNet152*

Berdasarkan grafik proses *training* pada gambar 6.19 menggunakan *ResNet152* mengalami kenaikan *accuracy* yang tidak stabil pada setiap epoch-nya. Pada data validasi di epoch 36 menuju epoch 37 juga sempat mengalami penurunan dari 0.71 ke 0.67. Namun pada epoch 39 menuju epoch 40 langsung mengalami kenaikan dari 0.62 ke 0.70. Kemudian sempat mengalami penurunan lagi pada epoch 46 menuju epoch 47 dari 0.82 ke 0.66. Hal itu terjadi berulang kali, begitu juga dengan data *training*, namun tidak terlalu naik turun seperti data validasi. Dan hasil akhir pada proses *training* ini berada di atas 80% yang berakhir pada epoch ke 50.



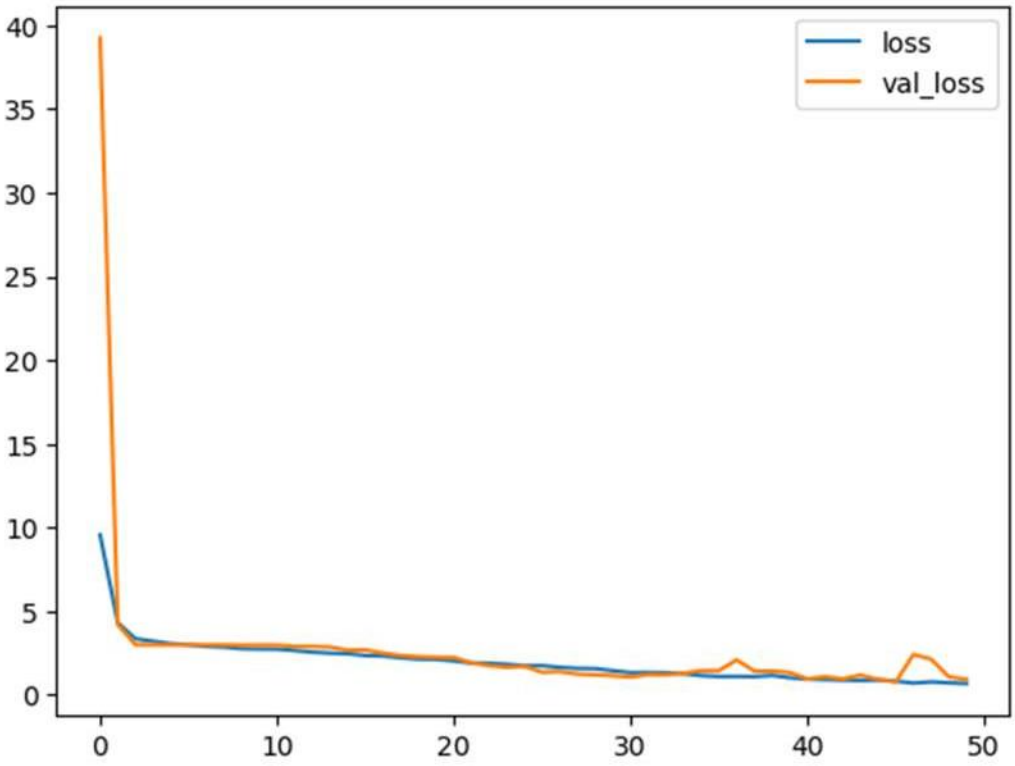
Gambar 43 Hasil Grafik *Training Loss* menggunakan *ResNet50*

Berdasarkan grafik proses *loss* pada gambar 6.20 menggunakan *ResNet50* mengalami penurunan *loss* yang tidak stabil. Pada data validasi di epoch 1 menuju epoch 2 mengalami penurunan drastis dari 9.8 ke 3.9, namun langsung mengalami peningkatan drastis pada epoch 2 menuju epoch 3 dari 3.9 ke 11.2. Setelah turun terus menerus hingga mengalami peningkatan lagi pada epoch 43 menuju epoch 44 dari 0.8 ke 3.7. Hal ini terjadi berulang kali hingga ke epoch 50. Begitu juga dengan data *training*, namun tidak terlalu naik turun seperti data validasi. Hasil akhir pada proses *training* ini berada di sekitar angka 1 dan akan mendekati 0 yang berakhir pada epoch ke 50.



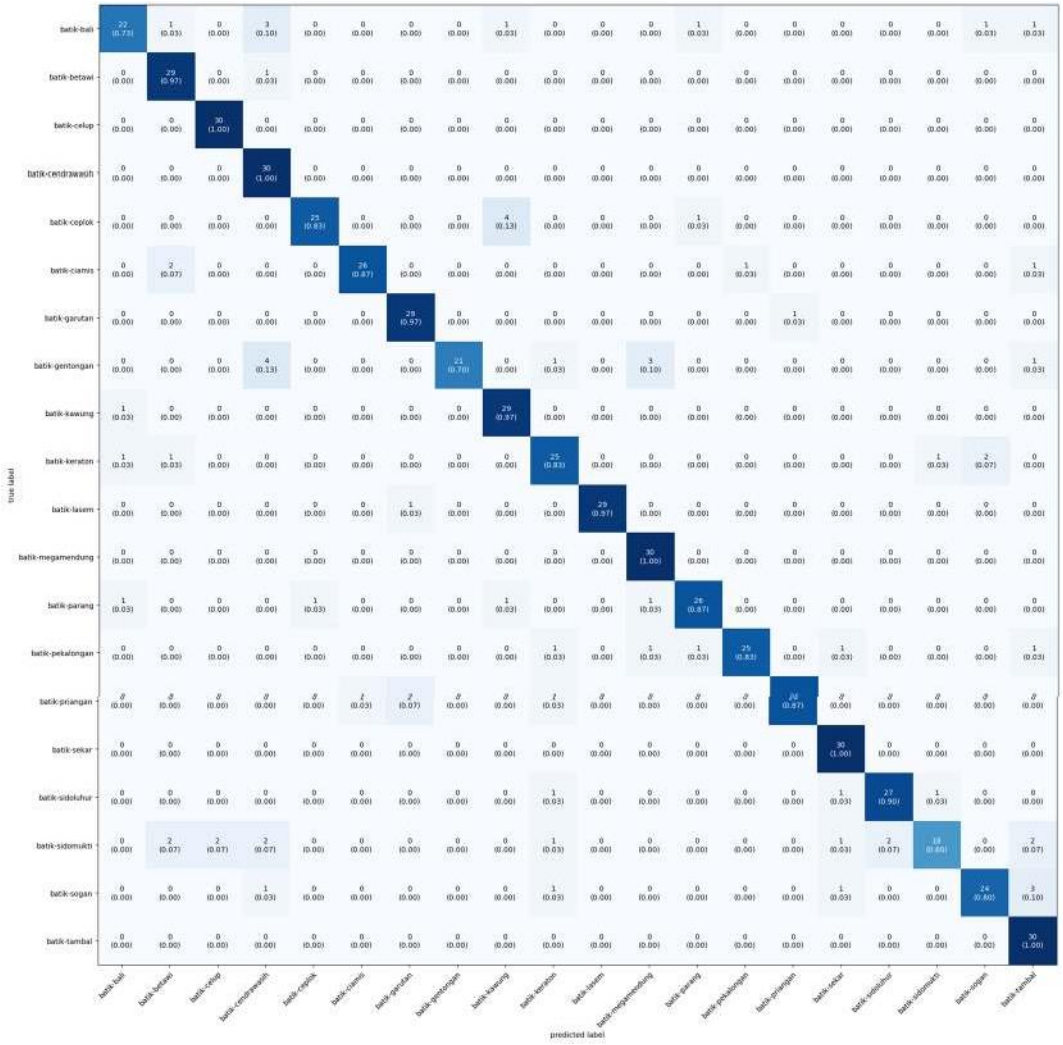
Gambar 44 Hasil Grafik *Training Loss* menggunakan *ResNet101*

Berdasarkan grafik proses *loss* pada gambar 6.21 menggunakan *ResNet101* mengalami penurunan *loss* yang tidak stabil. Pada data validasi di epoch 1 menuju epoch 2 mengalami penurunan drastis dari 92.8 ke 67.7, namun langsung mengalami peningkatan drastis juga pada epoch 3 menuju epoch 4 dari 11.4 ke 90.8. Kemudian langsung mengalami penurunan drastis lagi dari epoch 4 menuju epoch 5 dari 90.8 ke 6.2. Dan setelah turun terus menerus hingga epoch 50. Hasil akhir pada proses *training* ini berada di angka mendekati 0 yang berakhir pada epoch ke 50.



Gambar 45 Hasil Grafik *Training Loss* menggunakan *ResNet152*

Berdasarkan grafik proses *loss* pada gambar 6.22 menggunakan *ResNet152* mengalami penurunan *loss* yang tidak stabil. Pada data validasi di epoch 1 menuju epoch 2 mengalami penurunan drastis dari 39.2 ke 4.2. Setelah mengalami penurunan yang terus menerus, pada epoch 46 menuju epoch 47 sedikit mengalami kenaikan dari 0.7 ke 2.4. Dan setelah itu turun hingga epoch 50. Begitu juga dengan data *training*, namun tidak terlalu naik turun seperti data validasi. Hasil akhir pada proses *training* ini berada di angka mendekati 0 yang berakhir pada epoch ke 50.



Gambar 46 *Confusion Matrix* menggunakan *ResNet50*

Berdasarkan *confusion matrix* pada gambar 6.23 dapat dilihat bahwa proses *training* menggunakan *Resnet50* berhasil mengklasifikasikan dengan benar sebanyak 531 gambar. Berdasarkan hasil tersebut didapat hasil Batik Sidomukti yang mendapatkan jumlah terendah dalam pengklasifikasian menggunakan model ini. Batik Sidomukti hanya mendapatkan 18 gambar dari 30 gambar.

Pengujian ini dilakukan untuk mengukur presisi, sensitifitas, dan keakuratan dari data uji dan data latih terhadap hasil pengujian. Pengujian dilakukan dengan membandingkan antara hasil prediksi dengan hasil klasifikasi. Dari hasil *confusion matrix* ini bisa kita klasifikasikan sebagai berikut :

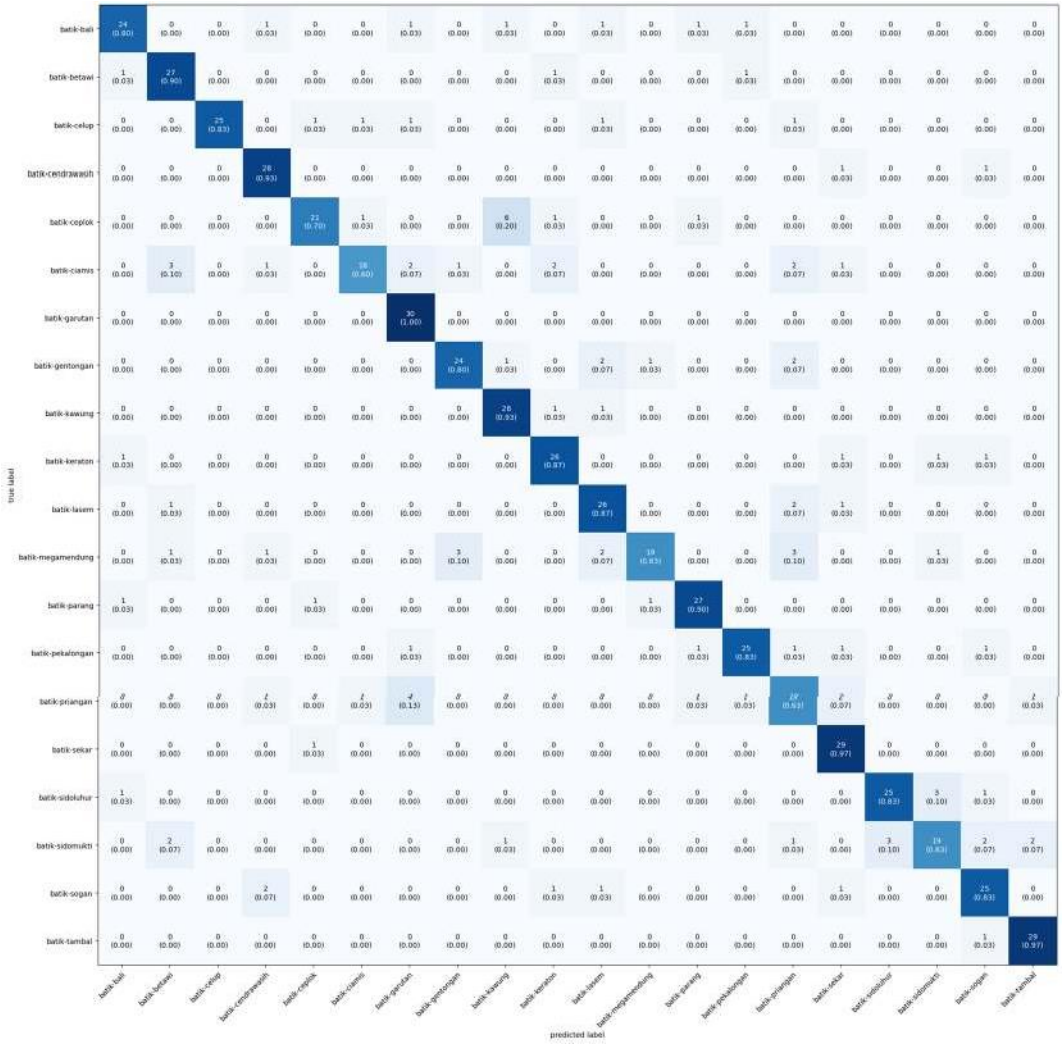
Tabel 18 Tabel Hasil Nilai dari Klasifikasi Data menggunakan Model *ResNet50*

Kelas	**Klasifikasi**				**Jumlah**
	TP	**FP**	**FN**	**TN**	
Batik Bali	22	8	3	567	600
Batik Betawi	29	1	6	564	600
Batik Celup	30	2	0	568	600
Batik Cendrawasih	30	0	11	559	600
Batik Ceplok	25	5	1	569	600
Batik Ciamis	26	4	1	569	600
Batik Garutan	29	1	3	567	600
Batik Gentongan	21	9	0	570	600
Batik Kawung	29	1	6	564	600
Batik Keraton	25	5	6	564	600
Batik Lasem	29	1	0	570	600
Batik Megamendung	30	0	5	565	600
Batik Parang	26	4	3	567	600
Batik Pekalongan	25	5	1	569	600
Batik Priangan	26	4	1	569	600
Batik Sekar	30	0	4	566	600
Batik Sidoluhur	27	3	2	568	600
Batik Sidomukti	18	12	2	568	600
Batik Sogan	24	6	3	567	600
Batik Tambal	30	0	9	561	600

Berdasarkan data klasifikasi pada Tabel 6.12 Klasifikasi Data dapat dihitung nilai presisi, sensitifitas, dan akurasinya sebagai berikut :

Tabel 19 Tabel Hasil Nilai dari Klasifikasi Data menggunakan Model *ResNet50*

Kelas	**Hasil**			
	Accuracy	***Recall***	***Precision***	***F1-Score***
Batik Bali	0,981667	0,733333	0,88	0,8
Batik Betawi	0,988333	0,966667	0,828571	0,892308
Batik Celup	0,996667	0,9375	1	0,967742
Batik Cendrawasih	0,981667	1	0,731707	0,84507
Batik Ceplok	0,99	0,833333	0,961538	0,892857
Batik Ciamis	0,991667	0,866667	0,962963	0,912281
Batik Garutan	0,993333	0,966667	0,90625	0,935484
Batik Gentongan	0,985	0,7	1	0,823529
Batik Kawung	0,988333	0,966667	0,828571	0,892308
Batik Keraton	0,981667	0,833333	0,806452	0,819672
Batik Lasem	0,998333	0,966667	1	0,983051
Batik Megamendung	0,991667	1	0,857143	0,923077
Batik Parang	0,988333	0,866667	0,896552	0,881356
Batik Pekalongan	0,99	0,833333	0,961538	0,892857
Batik Priangan	0,991667	0,866667	0,962963	0,912281
Batik Sekar	0,993333	1	0,882353	0,9375
Batik Sidoluhur	0,991667	0,9	0,931034	0,915254
Batik Sidomukti	0,976667	0,6	0,9	0,72
Batik Sogan	0,985	0,8	0,888889	0,842105
Batik Tambal	0,985	1	0,769231	0,869565



Gambar 47 *Confusion Matrix* menggunakan *ResNet101*

Berdasarkan *confusion matrix* pada gambar 6.24 dapat dilihat bahwa proses *training* menggunakan *ResNet101* berhasil mengklasifikasikan dengan benar sebanyak 494 gambar. Berdasarkan hasil tersebut didapat hasil Batik Ciamis yang mendapatkan jumlah terendah dalam pengklasifikasian menggunakan model ini. Batik Ciamis hanya mendapatkan 18 gambar dari 30 gambar.

Pengujian ini dilakukan untuk mengukur presisi, sensitifitas, dan keakuratan dari data uji dan data latih terhadap hasil pengujian. Pengujian dilakukan dengan membandingkan antara hasil prediksi dengan hasil klasifikasi. Dari hasil *confusion matrix* ini bisa kita klasifikasikan sebagai berikut :

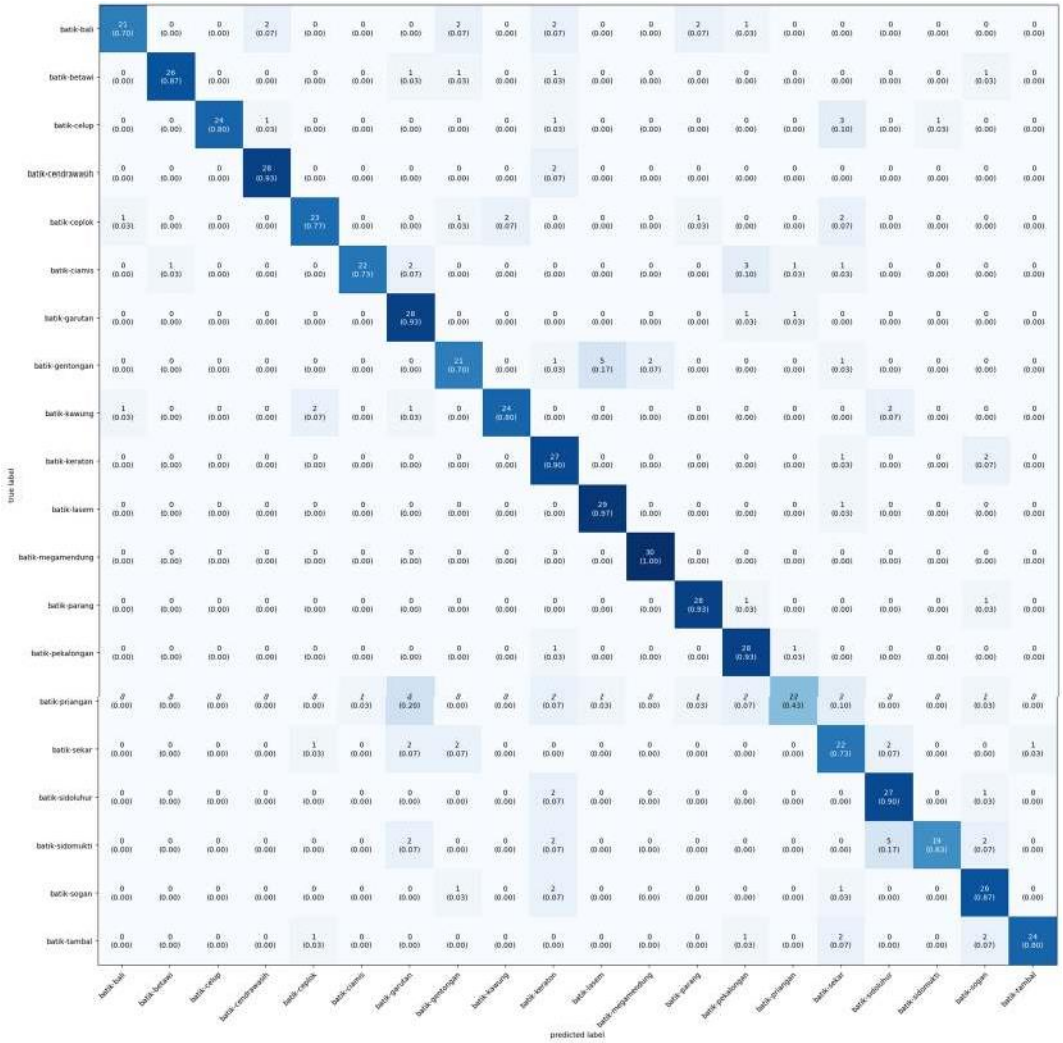
Tabel 20 Hasil Nilai dari Klasifikasi Data menggunakan Model *ResNet101*

Kelas	**Klasifikasi**				**Jumlah**
	TP	**FN**	**FP**	**TN**	
Batik Bali	26	5	25	544	600
Batik Betawi	26	4	17	553	600
Batik Celup	20	4	10	566	600
Batik Cendrawasih	22	8	2	568	600
Batik Ceplok	23	7	6	564	600
Batik Ciamis	24	6	23	547	600
Batik Garutan	19	11	6	564	600
Batik Gentongan	18	11	4	567	600
Batik Kawung	26	4	6	564	600
Batik Keraton	19	11	0	570	600
Batik Lasem	10	20	0	570	600
Batik Megamendung	16	14	1	569	600
Batik Parang	27	3	29	541	600
Batik Pekalongan	20	10	3	567	600
Batik Priangan	16	14	2	568	600
Batik Sekar	19	11	0	570	600
Batik Sidoluhur	20	10	9	561	600
Batik Sidomukti	22	8	2	568	600
Batik Sogan	24	6	4	566	600
Batik Tambal	28	2	32	538	600

Berdasarkan data klasifikasi pada Tabel 6.14 Klasifikasi Data dapat dihitung nilai presisi, sensitifitas, dan akurasinya sebagai berikut:

Tabel 21 Tabel Hasil Nilai dari Klasifikasi Data menggunakan Model *ResNet101*

Kelas	**Hasil**			
	Accuracy	***Recall***	***Precision***	***F1-Score***
Batik Bali	0,95	0,83871	0,509804	0,634146
Batik Betawi	0,965	0,866667	0,604651	0,712329
Batik Celup	0,976667	0,833333	0,666667	0,740741
Batik Cendrawasih	0,983333	0,733333	0,916667	0,814815
Batik Ceplok	0,978333	0,766667	0,793103	0,779661
Batik Ciamis	0,951667	0,8	0,510638	0,623377
Batik Garutan	0,971667	0,633333	0,76	0,690909
Batik Gentongan	0,975	0,62069	0,818182	0,705882
Batik Kawung	0,983333	0,866667	0,8125	0,83871
Batik Keraton	0,981667	0,633333	1	0,77551
Batik Lasem	0,966667	0,333333	1	0,5
Batik Megamendung	0,975	0,533333	0,941176	0,680851
Batik Parang	0,946667	0,9	0,482143	0,627907
Batik Pekalongan	0,978333	0,666667	0,869565	0,754717
Batik Priangan	0,973333	0,533333	0,888889	0,666667
Batik Sekar	0,981667	0,633333	1	0,77551
Batik Sidoluhur	0,968333	0,666667	0,689655	0,677966
Batik Sidomukti	0,983333	0,733333	0,916667	0,814815
Batik Sogan	0,983333	0,8	0,857143	0,827586
Batik Tambal	0,943333	0,933333	0,466667	0,622222



Gambar 48 *Confusion Matrix* menggunakan *ResNet152*

Berdasarkan *confusion matrix* pada gambar 6.25 dapat dilihat bahwa proses *training* menggunakan *ResNet152* berhasil mengklasifikasikan dengan benar sebanyak 490 gambar. Berdasarkan hasil tersebut didapat hasil Batik Priangan yang mendapatkan jumlah terendah dalam pengklasifikasian menggunakan model ini. Batik Ciamis hanya mendapatkan 13 gambar dari 30 gambar.

Pengujian ini dilakukan untuk mengukur presisi, sensitifitas, dan keakuratan dari data uji dan data latih terhadap hasil pengujian. Pengujian dilakukan dengan membandingkan antara hasil prediksi dengan hasil klasifikasi. Dari hasil *confusion matrix* ini bisa kita klasifikasikan sebagai berikut :

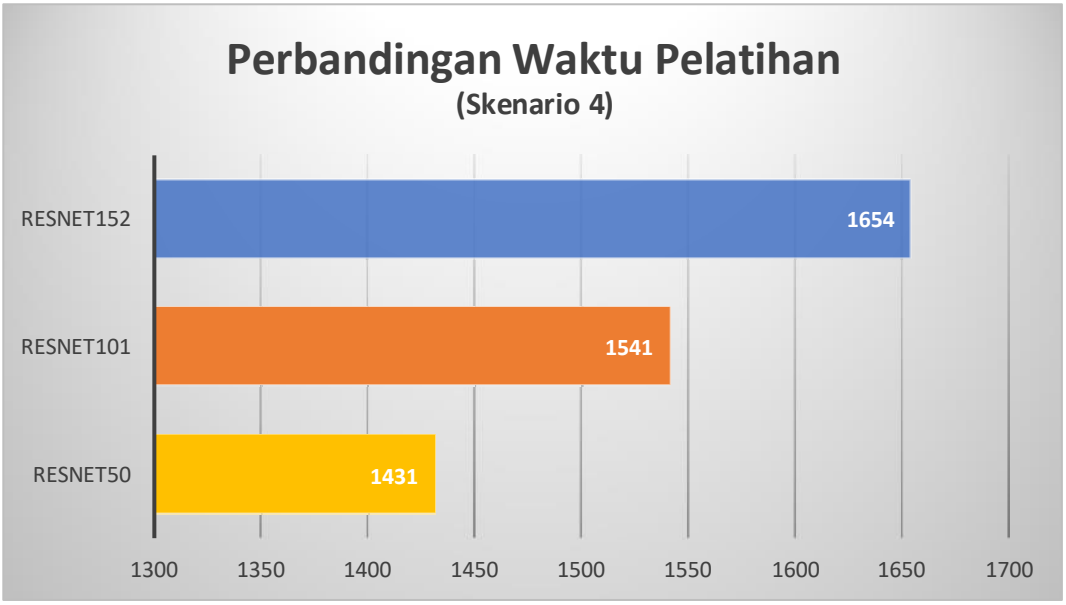
Tabel 22 Hasil Nilai dari Klasifikasi Data menggunakan Model *ResNet152*

Kelas	**Klasifikasi**				**Jumlah**
	TP	**FN**	**FP**	**TN**	
Batik Bali	21	9	2	568	600
Batik Betawi	26	4	1	569	600
Batik Celup	24	0	6	570	600
Batik Cendrawasih	28	2	3	567	600
Batik Ceplok	23	7	4	566	600
Batik Ciamis	22	8	1	569	600
Batik Garutan	28	2	14	556	600
Batik Gentongan	21	9	7	563	600
Batik Kawung	24	6	2	568	600
Batik Keraton	27	3	16	554	600
Batik Lasem	29	1	6	564	600
Batik Megamendung	30	0	2	568	600
Batik Parang	28	2	4	566	600
Batik Pekalongan	28	2	9	561	600
Batik Priangan	13	17	3	567	600
Batik Sekar	22	8	15	555	600
Batik Sidoluhur	27	3	9	561	600
Batik Sidomukti	19	11	1	569	600
Batik Sogan	26	4	10	560	600
Batik Tambal	24	6	1	569	600

Berdasarkan data klasifikasi pada Tabel 6.16, Klasifikasi Data dapat dihitung nilai presisi, sensitifitas, dan akurasinya sebagai berikut:

Tabel 23 Tabel Hasil Nilai dari Klasifikasi Data menggunakan Model *ResNet152*

Kelas	**Hasil**			
	Accuracy	***Recall***	***Precision***	***F1-Score***
Batik Bali	0,981667	0,7	0,913043	0,792453
Batik Betawi	0,991667	0,866667	0,962963	0,912281
Batik Celup	0,99	1	0,8	0,888889
Batik Cendrawasih	0,991667	0,933333	0,903226	0,918033
Batik Ceplok	0,981667	0,766667	0,851852	0,807018
Batik Ciamis	0,985	0,733333	0,956522	0,830189
Batik Garutan	0,973333	0,933333	0,666667	0,777778
Batik Gentongan	0,973333	0,7	0,75	0,724138
Batik Kawung	0,986667	0,8	0,923077	0,857143
Batik Keraton	0,968333	0,9	0,627907	0,739726
Batik Lasem	0,988333	0,966667	0,828571	0,892308
Batik Megamendung	0,996667	1	0,9375	0,967742
Batik Parang	0,99	0,933333	0,875	0,903226
Batik Pekalongan	0,981667	0,933333	0,756757	0,835821
Batik Priangan	0,966667	0,433333	0,8125	0,565217
Batik Sekar	0,961667	0,733333	0,594595	0,656716
Batik Sidoluhur	0,98	0,9	0,75	0,818182
Batik Sidomukti	0,98	0,633333	0,95	0,76
Batik Sogan	0,976667	0,866667	0,722222	0,787879
Batik Tambal	0,988333	0,8	0,96	0,872727



Gambar 49 Perbandingan Waktu yang Digunakan untuk Pelatihan pada Skenario 4

Pada gambar 6.28 menunjukkan bahwa pelatihan menggunakan model *ResNet50* mendapatkan waktu yang lebih cepat daripada menggunakan model yang lainnya. Hal ini dikarenakan perbedaan pada jumlah layer di setiap modelnya.

Berdasarkan uji coba membandingkan model dengan menggunakan *ResNet50* hingga *ResNet152* menunjukkan bahwa percobaan menggunakan model *ResNet50* memiliki nilai yang lebih tinggi daripada model yang lainnya. Perbandingan ini bisa dilihat pada tabel 6.18

Tabel 24 Tabel classification metrics skenario 5

Model	*Accuracy*	*Precision*	*Recall*	*F1-score*
ResNet50	0.89	0.89	0.89	0.88
ResNet101	0.82	0.83	0.82	0.82
ResNet152	0.82	0.84	0.82	0.82

Pada jaringan ResNet 50, 101 dan 152 layer, terdapat modifikasi pada blok bangunan koneksi pintas. Modifikasi tersebut adalah setiap fungsi residual ditumpuk 3 lapisan konvolusi dengan ukuran 1x1, 3x3, dan 1x1. Modifikasi ini disebut Deeper Bottleneck architecture, dimana lapisan terakhir 1x1 yang responsible untuk mengurangi dan meningkatkan dimensi dengan meninggakan lapisan konvolusi 3x3 input/output yang lebih kecil.

Pada ResNet 50, 101, dan 152 layer digunakan pintasan identitas yang bebas parameter (parameter-free identity shortcut).

Pada ResNet 50 layer, untuk meningkatkan dimensi menggunakan projection shortcut pada pintasan identitas (identity shortcut) dan mengubah 2 layer block pada ResNet 34 layer dengan 3 layer bottleneck block. Sedangkan pada ResNet 101 dan 152 layer menggunakan lebih dari 3 layer block. Meskipun ResNet 152 layer kedalamannya meningkat secara signifikan, namun masih mempunyai kompleksitas yang lebih rendah dari ResNet50. Model yang dibuat menggunakan hasil ekstraksi fitur gambar dari ResNet50. Nilai input model diambil dari vektor terakhir dari proses ekstraksi fitur.

Meskipun arsitektur ResNet101 dan ResNet152 lebih kompleks dari ResNet50 namun jika dilihat dari hasil evaluasi model, ResNet101 memiliki performa yang lebih buruk. Hal ini diakibatkan oleh data latih yang relatif kecil apabila dilatih dengan arsitektur yang besar seperti ResNet-101, belum tentu menghasilkan akurasi yang lebih baik.

BAB VII. KESIMPULAN DAN SARAN

7.1. Kesimpulan

Dari proses penelitian untuk skripsi peneliti yang berjudul "Implementasi *Image Classification* Menggunakan Arsitektur *Convolutional Neural Network*". Maka dapat diambil kesimpulan bahwasanya peneliti telah berhasil merancang dan membangun model pelatihan sesuai dengan rancangan. Dalam penelitian tersebut dapat diambil kesimpulan sebagai berikut:

1. Penelitian menggunakan arsitektur *Convolutional Neural Network* mendapatkan hasil sebesar 95%
2. Pelatihan model yang mendapatkan akurasi tertinggi yaitu sebesar 88% menggunakan model *ResNet50* dengan optimizer RMSprop dengan *learing rate* sebesar 0.0001.
3. Model paling optimal mendapatkan akurasi sebesar 88% menggunakan model *ResNet50* dengan optimizer RMSprop dengan *learing rate* sebesar 0.0001. Model ini berhasil mengklasifikasikan 531 dari 600 gambar batik dengan kelas yang benar dengan waktu pelatihan selama 5458 detik.
4. Semakin kecil nilai *learning rate*, semakin kecil kemungkinan terjadinya *overfitting* dan mendapatkan akurasi yang lebih baik.
5. Semakin tinggi kita menggunakan model pelatihan *ResNet* semakin lama juga durasi yang dilakukan ketika pelatihan, dan akurasi yang didapat juga tidak cukup baik. Hal ini dikarenakan jumlah layer yang semakin banyak.

7.2. Saran

Saran yang dapat diberikan untuk pengembangan lebih lanjut dari penelitian yang berjudul "Implementasi *Image Classification* Menggunakan Arsitektur *Convolutional Neural Network*" adalah sebagai berikut.

1. Menambah citra pada dataset di setiap kelasnya. Semakin banyak citra pada dataset, mesin akan mempelajari dan memproses citra tersebut hingga mendapatkan nilai akurasi yang lebih tinggi lagi
2. Merancang tambahan layer dengan lebih sederhana, sehingga ketika pada proses pelatihan tidak membutuhkan waktu yang lama.

DAFTAR PUSTAKA

Anggraini, W. (n.d.). DEEP LEARNING UNTUK DETEKSI WAJAH YANG BERHIJAB MENGGUNAKAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK (CNN) DENGAN TENSORFLOW. *2020*.

Falahkhi, B., Achmal, E. F., Rizaldi, M., Rizki, R., & Yudistira, N. (n.d.). Perbandingan Model AlexNet dan ResNet dalam Klasifikasi Citra Bunga Memanfaatkan Transfer Learning Comparison of AlexNet and ResNet Models in Flower Image Classification Utilizing Transfer Learning. *2022*. http://journal.ipb.ac.id/index.php/jika

Hadianto, N., Novitasari, H. B., & Rahmawati, A. (2019). KLASIFIKASI PEMINJAMAN NASABAH BANK MENGGUNAKAN METODE NEURAL NETWORK. *Jurnal Pilar Nusa Mandiri*, *15*(2), 163–170. https://doi.org/10.33480/pilar.v15i2.658

Hasan Mahmud, K., & al Faraby, S. (2019). *Klasifikasi Citra Multi-Kelas Menggunakan Convolutional Neural Network*.

Homepage, J., Roihan, A., Abas Sunarya, P., & Rafika, A. S. (2019). IJCIT (Indonesian Journal on Computer and Information Technology) Pemanfaatan Machine Learning dalam Berbagai Bidang: Review paper. In *IJCIT (Indonesian Journal on Computer and Information Technology)* (Vol. 5, Issue 1).

Kholik, A. (2021). KLASIFIKASI MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK (CNN) PADA TANGKAPAN LAYAR HALAMAN INSTAGRAM. *JDMSI*, *2*(2), 10–20.

Kusumanto, R. D., & Tompunu, A. N. (2011). PENGOLAHAN CITRA DIGITAL UNTUK MENDETEKSI OBYEK MENGGUNAKAN PENGOLAHAN WARNA MODEL NORMALISASI RGB. In *Seminar Nasional Teknologi Informasi & Komunikasi Terapan*.

Mawan, R. (2020). Klasifikasi motif batik menggunakan Convolutional Neural Network. *Jnanaloka*, 45–50. https://doi.org/10.36802/jnanaloka.2020.v1-no1-45-50

Pengolahan…, A., Zaid Munantri, N., Sofyan, H., & Yanu, M. (2019). APLIKASI PENGOLAHAN CITRA DIGITAL UNTUK IDENTIFIKASI UMUR POHON. In *TELEMATIKA* (Vol. 16, Issue 2).

Triyani, E., Harsono, D., & Sulistyowati, R. (2019). IJCIT (Indonesian Journal on Computer and Information Technology). In *IJCIT (Indonesian Journal on Computer and Information Technology)* (Vol. 5, Issue 1).

Tumewu, S. F., Setiabud, D. H., & Sugiarto, I. (2020). Klasifikasi Motif Batik menggunakan Metode Deep Convolutional Neural Network dengan Data Augmentation. *Jurnal Infra*, *8*(2), 189–194.

Wei, M., Wu, Q., Ji, H., Wang, J., Lyu, T., Liu, J., & Zhao, L. (2023). A Skin Disease Classification Model Based on DenseNet and ConvNeXt Fusion. *Electronics (Switzerland)*, *12*(2). https://doi.org/10.3390/electronics12020438