KLASIFIKASI MOTIF BATIK KHAS SEMARANG MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK*

SKRIPSI

Diajukan untuk Memenuhi Sebagian Syarat Guna Memperoleh
Gelar Sarjana Strata Satu (S-1)
dalam Ilmu Teknologi Informasi

Diajukan Oleh:

RIZKY NUR ARIFIN

NIM : 2108096094

PROGRAM STUDI TEKNOLOGI INFORMASI

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS ISLAM NEGERI WALISONGO SEMARANG

2024

PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama : Rizky Nur Arifin

NIM : 2108096094

Jurusan : Teknologi Informasi

Menyatakan bahwa skripsi berjudul:

KLASIFIKASI MOTIF BATIK KHAS SEMARANG MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK*

Secara keseluruhan adalah hasil penelitian/karya saya sendiri, kecuali bagian tertentu yang dirujuk sumbernya.

Semarang, 23 April 2025

Pembuat Pernyataan,

Rizky Nur Arifin

NIM. 2108096094

PENGESAHAN

KEMENTERIAN AGAMA
UNIVERSITAS ISLAM NEGERI WALISONGO
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Prof Dr. Hamka Kampus III Ngaliyan Semarang
Telp. (024) 7604554 Fax. 7615387

LEMBAR PENGESAHAN

Naskah skripsi berikut ini:

Judul	: Klasifikasi Motif Batik Khas Semarang Menggunakan *Convolutional Neural Network*
Penulis	: Rizky Nur Arifin
NIM	: 2108096094
Jurusan	: Teknologi Informasi

Telah diujikan dalam sidang tugas akhir oleh Dewan Penguji Fakultas Sains dan Teknologi UIN Walisongo Semarang dan dapat diterima sebagai salah satu syarat memperoleh gelar sarjana dalam Ilmu Teknologi Informasi.

Semarang, 2 Juni 2025

DEWAN PENGUJI

Ketua Sidang	Sekretaris Sidang
Dr. Masy Ari Ulinuha, M.T	Hery Mustofa, M.Kom
NIP. 19810812201101107	NIP. 198703172019031007

Penguji Utama I	Penguji Utama II
Dr. Wenty Dwi Yuniarti, M.Kom	Mokhamad Iklil Mustofa, M.Kom
NIP. 197706222006042005	NIP. 198808072019031010

Pembimbing I	Pembimbing II
Hery Mustofa, M.Kom	Dr. Khotibul Umam, M.Kom
NIP. 198703172019031007	NIP. 197908272011011007

NOTA PEMBIMBING

Semarang, 23 April 2025

Yth. Ketua Program Studi Teknologi Informasi
Fakultas Sains dan Teknologi

UIN Walisongo Semarang

Assalamu'alaikum Wr. Wb.

Dengan ini diberitahukan bahwa saya telah melakukan bimbingan, arahan, dan koreksi naskah skripsi dengan:

Judul : Klasifikasi Motif Batik Khas Semarang Menggunakan *Convolutional Neural Network*

Penulis : **Rizky Nur Arifin**

NIM : 2108096094

Jurusan : Teknologi Informasi

Saya memandang bahwa naskah skripsi tersebut sudah dapat diajukan kepada Fakultas Sains dan Teknologi UIN Walisongo Semarang untuk diujikan dalam Sidang Munaqosah.

Wassalamu'alaikum Wr. Wb.

Pembimbing I,

Hery Mustofa, M. Kom

NIP. 198703172019031007

NOTA PEMBIMBING

Semarang, 23 April 2025

Yth. Ketua Program Studi Teknologi Informasi
Fakultas Sains dan Teknologi
UIN Walisongo Semarang

Assalamu'alaikum Wr. Wb.

Dengan ini diberitahukan bahwa saya telah melakukan bimbingan, arahan, dan koreksi naskah skripsi dengan:

Judul : Klasifikasi Motif Batik Khas Semarang Menggunakan *Convolutional Neural Network*

Penulis : **Rizky Nur Arifin**

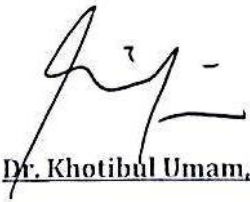
NIM : 2108096094

Jurusan : Teknologi Informasi

Saya memandang bahwa naskah skripsi tersebut sudah dapat diajukan kepada Fakultas Sains dan Teknologi UIN Walisongo Semarang untuk diujikan dalam Sidang Munaqosah.

Wassalamu'alaikum Wr. Wb.

Pembimbing II,



Dr. Khotibul Umam, S.T., M. Kom

NIP. 197908272011011007

LEMBAR PERSEMBAHAN

Dengan rasa syukur yang mendalam atas selesainya skripsi ini, penulis mempersembahkan karya ini kepada:

1. Keluarga besar penulis yang senantiasa memberikan dukungan dan doa.
2. Segenap civitas akademika UIN Walisongo Semarang, semoga senantiasa diberikan kesehatan dan semangat dalam menjalani aktivitas di kampus tercinta ini.
3. Teman-teman penulis yang selalu memberikan dukungan dan semangat.

MOTO

اِنَّ اللّٰهَ لَا يُغَيِّرُ مَا بِقَوْمٍ حَتّٰى يُغَيِّرُوْا مَا بِاَنْفُسِهِمْۗ

“Sesungguhnya Allah tidak mengubah keadaan suatu kaum hingga mereka mengubah apa yang ada pada diri mereka.”

(QS. Ar-Ra’d: 11)

KLASIFIKASI MOTIF BATIK KHAS SEMARANG

MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK*

Oleh:
Rizky Nur Arifin
2108096094

ABSTRAK

Batik merupakan salah satu warisan budaya Indonesia yang telah diakui oleh UNESCO sebagai Warisan Budaya Tak Benda sejak 2 Oktober 2009. Setiap daerah di Indonesia memiliki motif batik yang khas, termasuk batik Semarang yang dikenal dengan keunikan motifnya yang menggambarkan flora, fauna, serta simbol-simbol budaya khas Kota Semarang. Namun, masyarakat seringkali kesulitan mengenali dan membedakan motif batik Semarang karena keberagaman motif yang ada serta kurangnya pemahaman tentang motif-motif tersebut. Oleh karena itu, diperlukan sistem berbasis kecerdasan buatan yang dapat membantu masyarakat dalam mengenali dan membedakan motif batik Semarang. Penelitian ini bertujuan untuk mengembangkan sistem klasifikasi motif batik Semarang menggunakan *Convolutional Neural Network* (CNN) dengan pendekatan *transfer learning* berbasis arsitektur MobileNetV2. Model dikembangkan menggunakan dataset batik Semarang dan dikonversi ke dalam format *TensorFlow Lite* untuk diimplementasikan pada aplikasi Android. Evaluasi model menggunakan *confusion matrix* menunjukkan akurasi 100% pada data uji, yang mengindikasikan bahwa model dapat mengklasifikasikan batik Semarang dengan tepat. Selain itu, aplikasi Android berhasil dibangun untuk mengimplementasikan model ini, sehingga pengguna dapat mengklasifikasikan gambar batik secara langsung melalui perangkat.

Kata Kunci: Klasifikasi, Batik Semarang, *Convolutional Neural Network*, MobileNetV2, *Transfer Learning*

KATA PENGANTAR

Alhamdulillah, Puji Syukur atas kehadirat Allah SWT atas segala nikmat dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul "Klasifikasi Motif Batik Khas Semarang Menggunakan *Convolutional Neural Network*" dengan baik. Sholawat serta salam tercurahkan pada Nabi Muhammad SAW. Semoga syafaatnya mengalir kepada kita hingga hari akhir kelak, aamiin.

Penyusunan skripsi ini merupakan salah satu syarat untuk memperoleh gelar Sarjana Komputer pada Program Studi Teknologi Informasi di UIN Walisongo Semarang. Proses penyusunan ini tentu tidak terlepas dari dukungan dan do'a dari berbagai pihak. Oleh karena itu, izinkan penulis dengan tulus menyampaikan ucapan terima kasih kepada:

1. Bapak Prof. Dr. H. Musahadi, M.Ag, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Walisongo Semarang.
2. Bapak Dr. Khotibul Umam, S.T., M.Kom, selaku Ketua Program Studi Teknologi Informasi Universitas Islam Negeri Walisongo Semarang.
3. Bapak Hery Mustofa, M.Kom, dan Bapak Dr. Khotibul Umam, S.T., M.Kom, selaku dosen pembimbing yang telah memberikan arahan dan bimbingan selama proses penyusunan skripsi hingga selesai.

4. Seluruh staf, karyawan, dan dosen di lingkungan Universitas Islam Negeri Walisongo Semarang.
5. Orang tua tercinta dan keluarga yang senantiasa mendoakan serta memberikan dukungan baik secara moril maupun materil kepada penulis.
6. Teman-teman Program Studi Teknologi Informasi Angkatan 2021, khususnya kelas C, atas kebersamaan, kerja sama, dan dukungan yang telah diberikan.
7. Teman-teman Kos Kharisma atas kebersamaan, kerja sama, dan dukungan yang telah diberikan.
8. Semua pihak yang tidak dapat disebutkan satu per satu, namun tetap memberikan kontribusi dan dukungan yang berarti bagi penulis.

Penulis menyadari bahwa skripsi ini masih jauh dari kata sempurna dan terdapat banyak kekurangan. Oleh karena itu, kritik dan saran yang membangun sangat diharapkan demi perbaikan di masa mendatang. Semoga skripsi ini dapat memberikan manfaat bagi semua pihak yang berkepentingan. *Aamiin Yaa Rabbal 'Alamin*.

Semarang, 23 April 2025

Penulis,

Rizky Nur Arifin

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

A. Latar Belakang

Batik merupakan salah satu warisan budaya Indonesia yang telah diakui oleh UNESCO sebagai Warisan Budaya Tak Benda sejak 2 Oktober 2009 (Negara et al., 2021). Setiap daerah di Indonesia memiliki motif batik yang khas, termasuk batik Semarang yang dikenal dengan keunikan motifnya yang menggambarkan flora, fauna, serta simbol-simbol budaya khas Kota Semarang. Namun, masyarakat seringkali kesulitan mengenali dan membedakan motif batik Semarang karena keberagaman motif yang ada serta kurangnya pemahaman tentang motif-motif tersebut. Oleh karena itu, diperlukan sistem berbasis kecerdasan buatan yang dapat membantu masyarakat dalam mengenali dan membedakan motif batik Semarang.

Seiring perkembangan teknologi, khususnya dalam bidang kecerdasan buatan (*Artificial Intelligence*) dan visi komputer (*Computer Vision*), tantangan dalam pengenalan motif batik dapat diatasi melalui sistem berbasis *deep learning*. *Deep learning* merupakan cabang dari pembelajaran mesin (*machine learning*) yang meniru cara kerja otak manusia dalam memproses data (Putra et al., 2021). *Deep learning* menggunakan jaringan saraf tiruan

(*artificial neural networks*) dengan banyak lapisan (*deep neural networks*) untuk mempelajari fitur dari data secara bertahap (Negara et al., 2021). Keunggulan utama *deep learning* terletak pada kemampuannya untuk belajar secara mandiri dari data yang besar dan kompleks, menjadikannya salah satu pendekatan paling efektif dalam tugas seperti pengenalan gambar, pengenalan suara, dan penerjemahan bahasa.

Salah satu arsitektur *deep learning* yang umum digunakan dalam pengenalan gambar adalah *Convolutional Neural Network* (CNN) (Putra et al., 2021). CNN dirancang untuk mengenali pola visual dalam gambar melalui serangkaian lapisan konvolusi, yang memungkinkan model mengekstraksi fitur seperti tepi, tekstur, bentuk, dan warna secara otomatis. Dibandingkan dengan metode konvensional yang membutuhkan ekstraksi fitur secara manual, CNN mampu mempelajari pola dari data secara otomatis, menjadikannya lebih efisien dalam tugas pengenalan dan klasifikasi gambar . Kemampuannya ini membuat CNN banyak digunakan dalam berbagai penelitian terkait klasifikasi gambar, termasuk dalam klasifikasi motif batik dari berbagai daerah di Indonesia (Uswatun Khasanah et al., 2020).

Seiring dengan meningkatnya variasi motif batik dari berbagai daerah, diperlukan sistem yang tidak hanya memiliki akurasi tinggi dalam mengenali motif, tetapi juga efisien dalam proses komputasi. Salah satu arsitektur CNN yang dirancang untuk mencapai efisiensi tersebut adalah MobileNetV2. MobileNetV2 merupakan pengembangan dari arsitektur MobileNet sebelumnya, dengan *depthwise separable convolutions* untuk mengurangi jumlah parameter, serta *inverted residuals* dan *linear bottlenecks* untuk mempertahankan akurasi meskipun modelnya ringan (Sandler et al., 2018). Hal ini membuat MobileNetV2 cocok diterapkan pada perangkat dengan sumber daya terbatas seperti smartphone (Sastypratiwi et al., 2024).

Sebagai model yang dirancang ringan dengan tetap mempertahankan akurasi, MobileNetV2 tetap memerlukan data pelatihan yang cukup untuk mencapai performa optimal. Namun, pengumpulan dataset dalam jumlah besar sering menjadi tantangan karena keterbatasan akses dan ketersediaan data berkualitas. Untuk mengatasi hal tersebut, diterapkan *transfer learning*, yang memungkinkan model memanfaatkan bobot dari jaringan yang telah dilatih pada dataset besar seperti ImageNet, sehingga dapat mengenali pola lebih cepat dan akurat meskipun dengan dataset yang kecil (Putra et al., 2021).

Penggunaan teknologi dalam penelitian ini dapat dilihat sebagai bagian dari upaya untuk terus mengembangkan ilmu pengetahuan, sebagaimana Islam mendorong umatnya untuk terus menuntut ilmu. Dalam QS. Al-'Alaq: 1-5, Allah SWT berfirman:

اِقْرَأْ بِاسْمِ رَبِّكَ الَّذِيْ خَلَقَۚ ﴿١﴾ خَلَقَ الْاِنْسَانَ مِنْ عَلَقٍۚ ﴿٢﴾ اِقْرَأْ
وَرَبُّكَ الْاَكْرَمُۙ ﴿٣﴾ الَّذِيْ عَلَّمَ بِالْقَلَمِۙ ﴿٤﴾ عَلَّمَ الْاِنْسَانَ مَا لَمْ يَعْلَمْۗ ﴿٥﴾

> *"Bacalah dengan (menyebut) nama Tuhanmu yang menciptakan, Dia telah menciptakan manusia dari segumpal darah. Bacalah, dan Tuhanmulah Yang Maha Pemurah, Yang mengajar (manusia) dengan perantaran kalam, Dia mengajar kepada manusia apa yang tidak diketahuinya." (QS. Al-Alaq ayat 1-5)*

Ayat ini menegaskan bahwa ilmu pengetahuan merupakan anugerah dari Allah SWT yang harus dikembangkan dan dimanfaatkan untuk kebaikan. Dalam konteks penelitian ini, penerapan teknologi kecerdasan buatan dalam klasifikasi motif batik adalah salah satu bentuk implementasi ilmu pengetahuan yang bertujuan untuk melestarikan budaya dan membantu masyarakat dalam mengenali warisan budaya Indonesia

B. Rumusan Masalah

Berdasarkan latar belakang sebelumnya, rumusan masalah dalam penelitian ini adalah sebagai berikut:

1. Bagaimana membangun model klasifikasi batik Semarang menggunakan MobileNetV2 dengan *transfer learning*?
2. Bagaimana mengevaluasi kinerja model yang dibangun dalam mengklasifikasikan batik Semarang?
3. Bagaimana membangun aplikasi Android untuk mengimplementasikan model klasifikasi motif batik Semarang?

C. Batasan Masalah

Agar penelitian lebih terarah, maka batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Dataset yang digunakan berasal dari Kaggle dengan 5 dari 10 kelas yang tersedia.
2. Model yang digunakan adalah MobileNetV2 dengan *transfer learning*, tanpa membandingkan dengan arsitektur CNN lainnya.
3. Evaluasi kinerja model dilakukan menggunakan *confusion matrix*.
4. Aplikasi Android dibangun menggunakan bahasa Kotlin.

D. Tujuan Penelitian

Berdasarkan rumusan masalah yang telah dijelaskan, tujuan penelitian ini adalah sebagai berikut:

1. Membangun model klasifikasi batik Semarang menggunakan MobileNetV2 dengan *transfer learning*.
2. Mengevaluasi kinerja model yang dibangun dalam mengklasifikasikan batik Semarang berdasarkan metrik evaluasi yang diperoleh dari *confusion matrix*.
3. Membangun aplikasi Android untuk mengimplementasikan model klasifikasi motif batik Semarang.

E. Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat sebagai berikut:

1. Manfaat Teoritis: Penelitian ini diharapkan dapat menjadi referensi bagi penelitian selanjutnya mengenai klasifikasi motif batik menggunakan *Convolutional Neural Network* (CNN).
2. Manfaat Praktis: Penelitian ini diharapkan dapat membantu masyarakat dalam mengidentifikasi dan mengenali motif batik Semarang melalui aplikasi yang dibangun.

BAB II
LANDASAN PUSTAKA

A. Kajian Teori

1. Batik

   Kata "batik" berasal dari bahasa Jawa, yaitu "amba" yang berarti lebar dan "nitik" yang berarti memberi titik. Dalam Kamus Besar Bahasa Indonesia, batik memiliki arti kain bergambar yang pembuatannya secara khusus dengan menuliskan atau menerakan malam pada kain itu, kemudian pengolahannya diproses dengan cara tertentu.

2. Batik Semarang

   Pada masa pemerintahan kolonial Hindia Belanda, ada dua kategori batik, yaitu batik Kerajaan atau Keraton dan batik Pesisir. Batik Keraton adalah batik yang diproduksi di wilayah kerajaan Kesunanan Surakarta (Solo) dan Kasultanan Yogyakarta. Batik Pesisir atau Pesisiran adalah batik yang diproduksi di luar wilayah keraton Surakarta dan Yogyakarta.

   Batik Semarang termasuk dalam kategori batik Pesisir karena letaknya yang berada di pesisir utara Jawa. Meskipun memiliki kemiripan motif dengan batik dari daerah pesisir utara Jawa, batik Semarang tidak sepopuler batik Pekalongan, Yogyakarta, atau Solo.

Kendati demikian, batik ini memiliki ciri khas tersendiri, ditandai dengan warna dasar oranye kemerahan dan motif naturalis yang menampilkan elemen flora dan fauna. Selain itu, saat ini motif batik Semarang juga telah mencakup berbagai ikon yang merepresentasikan kota dan budaya Semarang (Intan et al., 2023).

Batik Semarang memiliki berbagai macam motif. Berikut ini adalah beberapa di antaranya:

a. Batik Asam Arang

Gambar 2. 1 Batik Asam Arang

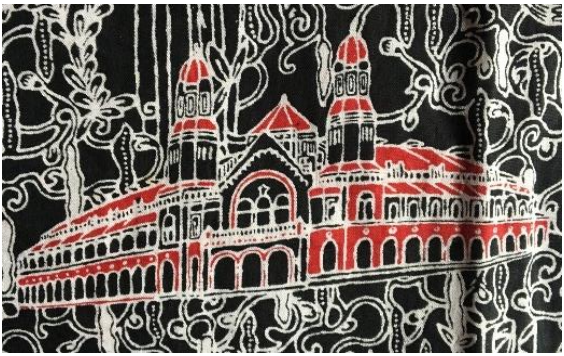
Kata "asam" merujuk pada pohon asam, sedangkan "arang" berarti jarang. Dengan demikian, motif ini melambangkan pohon asam yang tumbuh jarang, yang juga menjadi asal mula nama kota Semarang.

b. Batik Blekok Srondol

Gambar 2. 2 Batik Blekok Srondol

Motif ini terinspirasi dari fauna khas Semarang, yaitu burung blekok, yang banyak ditemukan di daerah Srondol, Semarang.

c. Batik Lawang Sewu



Gambar 2. 3 Batik Lawang Sewu

Motif ini terinspirasi dari bangunan Lawang Sewu, ikon bersejarah yang menjadi ciri khas Kota Semarang.

d. Batik Tugu Muda

Gambar 2. 4 Batik Tugu Muda

Motif ini terinspirasi dari monumen Tugu Muda di Semarang, yang dibangun untuk mengenang peristiwa pertempuran lima hari di kota tersebut.

e. Batik Warak Ngendog

Gambar 2. 5 Batik Warak Ngendog

Warak ngendog merupakan motif hewan yang dalam bahasa Jawa berarti “badak bertelur”.

Motif ini terinspirasi dari warak yang muncul dalam tradisi budaya Semarang, yaitu Dugderan, yang menandai dimulainya bulan suci Ramadan.

3. Klasifikasi Gambar

Klasifikasi gambar merupakan salah satu aplikasi dalam *deep learning* di mana komputer belajar mengenali dan mengidentifikasi kategori atau label dari suatu gambar berdasarkan konten visualnya. Sebagai bagian dari *computer vision*, yang merupakan bagian dari kecerdasan buatan, metode ini berfokus pada pengenalan dan pemahaman gambar oleh komputer.

Dalam konteks batik, klasifikasi dilakukan berdasarkan bentuk motifnya, dengan tujuan untuk mengelompokkan citra batik ke dalam kelas-kelas motif sesuai pola yang ada. Proses klasifikasi batik melibatkan serangkaian tahapan, mulai dari *preprocessing* hingga pengenalan menggunakan algoritma klasifikasi (Wiratama, 2023).

4. *Machine Learning*

Machine learning pertama kali diperkenalkan oleh Arthur Samuel, seorang pakar kecerdasan buatan pada tahun 1959. Samuel menjelaskan *machine learning* sebagai disiplin ilmu yang memungkinkan

komputer untuk belajar dan berkembang sendiri tanpa harus diprogram secara eksplisit untuk setiap tugas (Samuel, 1959). Dengan *machine learning*, komputer dapat mengenali pola dalam data, membuat prediksi, dan mengambil keputusan berdasarkan informasi yang diperoleh dari pengalaman atau data pelatihan.

Berikut adalah beberapa jenis *machine learning*:

a. *Supervised Learning*

   merupakan jenis *machine learning* di mana model dilatih menggunakan data yang sudah memiliki label.

b. *Unsupervised Learning*

   merupakan jenis *machine learning* di mana model dilatih menggunakan data yang tidak memiliki label.

c. *Semi-Supervised Learning*

   merupakan jenis *machine learning* di mana model dilatih menggunakan kombinasi data berlabel dan tidak berlabel.

d. *Reinforcement Learning*

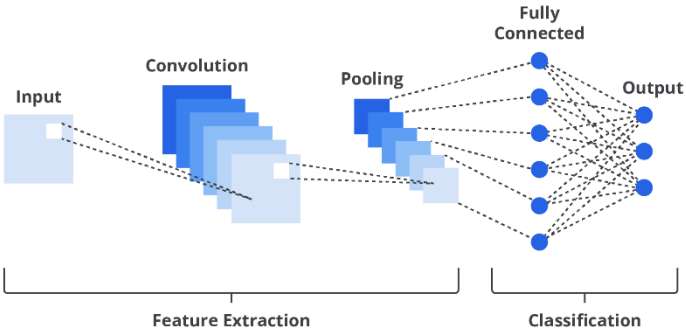
   merupakan jenis *machine learning* di mana agen belajar untuk mengambil keputusan dengan cara berinteraksi dengan lingkungan.

5. *Deep Learning*

   Deep learning merupakan cabang dari pembelajaran mesin (*machine learning*) yang meniru cara kerja otak manusia dalam memproses data (Putra et al., 2021). *Deep learning* menggunakan jaringan saraf tiruan (*artificial neural networks*) dengan banyak lapisan (*deep neural networks*) untuk mempelajari fitur dari data secara bertahap (Negara et al., 2021). Keunggulan utama *deep learning* terletak pada kemampuannya untuk belajar secara mandiri dari data yang besar dan kompleks, menjadikannya salah satu pendekatan paling efektif dalam tugas seperti pengenalan gambar, pengenalan suara, dan penerjemahan bahasa. Dalam pengenalan gambar, *Convolutional Neural Networks* (CNN) merupakan metode yang umum digunakan (Putra et al., 2021).

6. *Convolutional Neural Network*

   Convolutional Neural Network (CNN) merupakan algoritma *deep learning* yang merepresentasikan perkembangan *multi-layer perceptrons* (MLP) dan umum digunakan dalam pengenalan gambar (Putra et al., 2021).



Gambar 2. 6 Arsitektur CNN

Pada Gambar 2.6 menunjukkan terdapat dua tahapan utama dalam klasifikasi gambar menggunakan CNN, yakni *feature extraction* dan *classification*. Pada tahap *feature extraction*, gambar yang dimasukkan akan melalui beberapa lapisan konvolusi dan pooling yang berfungsi untuk mengekstrak fitur-fitur penting dari gambar, seperti tepi, tekstur, atau pola tertentu. Setiap lapisan CNN berfungsi untuk menangkap informasi yang semakin kompleks, mulai dari fitur yang sederhana hingga fitur yang lebih abstrak. Setelah fitur-fitur tersebut diekstraksi, gambar kemudian diteruskan ke tahap *classification*. Pada tahap ini, hasil ekstraksi fitur diproses lebih lanjut oleh lapisan *fully connected* yang mengubah representasi fitur tersebut menjadi vektor dan melakukan klasifikasi berdasarkan kelas yang telah dipelajari.

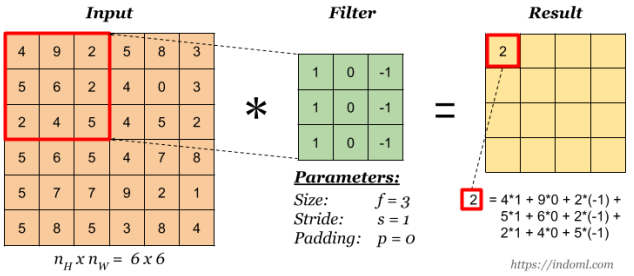
Convolutional Neural Network (CNN) terdiri dari tiga jenis lapisan utama, yaitu lapisan masukan (*input layer*), lapisan keluaran (*output layer*), dan lapisan tersembunyi (*hidden layer*). Pada lapisan tersembunyi, terdapat beberapa lapisan yang disusun bertingkat, seperti *convolutional layer*, *pooling layer*, dan *fully connected layer*. Berikut adalah penjelasan tentang lapisan-lapisan dalam arsitektur CNN:

a. *Input Layer*

Input layer merupakan lapisan awal dalam *neural network* yang berfungsi menerima data masukan sebelum diproses oleh jaringan. Jenis data yang diterima bervariasi sesuai dengan tugas yang dihadapi. Misalnya, dalam pengenalan gambar, data masukan berupa informasi piksel dari gambar.

b. *Convolutional Layer*

Convolutional layer merupakan lapisan pertama dalam CNN yang berfungsi untuk mengekstraksi fitur dari gambar masukan. Pada lapisan ini, filter (kernel) berukuran kecil bergerak melintasi gambar untuk menangkap fitur lokal, seperti tepi, sudut, atau tekstur. Hasil dari proses ini disebut *feature map*.

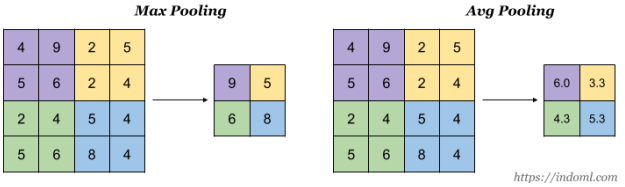


Gambar 2. 7 Convolutional Layer

Pada Gambar 2.7 menunjukkan sebuah matriks berukuran 6×6 yang dilakukan konvolusi menggunakan kernel/filter berukuran 3×3, dengan *stride* sebesar 1 dan *padding* nol (*zero padding*). Hasil konvolusi tersebut terlihat pada matriks *result*, yang sering disebut *feature map*.

c. *Pooling Layer*

Pooling layer merupakan lapisan dalam CNN yang berfungsi untuk mengurangi dimensi dari *feature map* yang dihasilkan oleh *convolutional layer*. Tujuannya adalah mengurangi jumlah parameter dan komputasi dalam jaringan, serta mencegah *overfitting* dan mempercepat proses pelatihan.



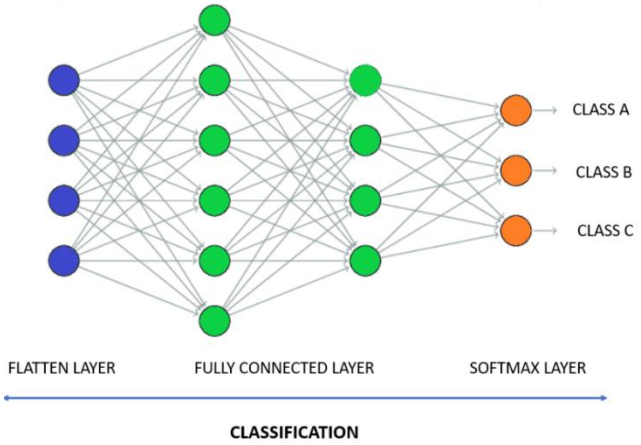
Gambar 2. 8 Pooling Layer

Pada Gambar 2.8 menunjukkan sebuah matriks berukuran 4×4 yang dilakukan *pooling* menggunakan filter berukuran 2×2, dengan *stride* sebesar 2. Pada proses *pooling* ini, dua metode yang umum digunakan adalah *max pooling* dan *average pooling*. Pada max pooling, nilai yang dipilih merupakan nilai maksimum dari setiap area filter 2×2. Sedangkan pada *average pooling*, nilai yang dipilih merupakan nilai rata-rata dari setiap area filter 2×2. Hasil *pooling* ini menghasilkan matriks berukuran 2×2, yang sering disebut *pooled feature map*.

d. Fully Connected Layer

Fully connected layer adalah bagian penting dari *neural network* yang biasanya terletak di bagian akhir arsitektur, setelah serangkaian convolutional dan pooling layer dalam CNN. Setiap neuron di lapisan ini terhubung ke semua neuron sebelumnya, dengan bobot yang dipelajari selama

pelatihan. Lapisan ini berfungsi untuk mempelajari pola dari fitur yang telah diekstraksi sebelum diteruskan ke *output layer* untuk menghasilkan prediksi akhir.



Gambar 2. 9 Fully Connected Layer

Pada Gambar 2.9 menunjukkan sebelum *fully connected layer*, data akan diproses terlebih dahulu di *flatten layer*. Proses *flatten* ini mengubah data yang awalnya berupa matriks hasil ekstraksi fitur dari *convolutional* dan *pooling layer* menjadi vektor satu dimensi. Vektor yang dihasilkan kemudian diteruskan ke *fully connected layer*. Setelah itu, output dari *fully connected layer* diteruskan ke *softmax* atau *output layer*, yang mengubahnya menjadi probabilitas untuk setiap kelas.

e. Output Layer

Output layer adalah lapisan terakhir dalam *neural network* yang berfungsi menghasilkan prediksi berdasarkan informasi yang telah diproses oleh lapisan-lapisan sebelumnya. Dalam tugas klasifikasi, lapisan ini biasanya memiliki jumlah neuron yang sesuai dengan jumlah kelas yang ada, serta menggunakan fungsi aktivasi seperti *softmax* untuk klasifikasi multi-kelas atau *sigmoid* untuk klasifikasi biner. Fungsi ini mengubah nilai output menjadi probabilitas, di mana kelas dengan probabilitas tertinggi dianggap sebagai prediksi akhir dari model.

f. Activation Function

Activation function adalah fungsi yang digunakan untuk menentukan nilai keluaran dari suatu neuron. Tanpa fungsi ini, *neural network* hanya dapat menghasilkan fungsi linear. Secara umum, terdapat dua jenis fungsi aktivasi: linear dan non-linear. Fungsi aktivasi linear biasanya digunakan dalam kasus regresi, sedangkan fungsi non-linear lebih umum untuk klasifikasi dan deteksi.

Berikut beberapa fungsi aktivasi non-linear yang umum digunakan dalam *CNN* (Akhiar, 2021):

1) Sigmoid

   Sigmoid adalah fungsi aktivasi yang umum digunakan untuk klasifikasi dua kelas. Fungsi ini mengubah input menjadi output dalam rentang 0 hingga 1. Berikut adalah persamaan fungsi aktivasi *sigmoid*:

$$f(x) = \frac{1}{1+e^{-x}} \qquad (2.1)$$

2) Tanh

   Tanh adalah fungsi aktivasi yang mengubah input menjadi output dalam rentang antara -1 dan 1. Fungsi ini efektif untuk data yang terdistribusi di sekitar nol. Berikut adalah persamaan fungsi aktivasi *tanh*:

$$f(x) = tanhx\,\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \qquad (2.2)$$

3) ReLU (*Rectified Linear Unit*)

   ReLU adalah fungsi aktivasi yang memberikan output nol untuk nilai input negatif dan output yang sama dengan input untuk nilai positif. Berikut adalah persamaan fungsi aktivasi *ReLU*:

$$f(x) = \max(0, x) \qquad (2.3)$$

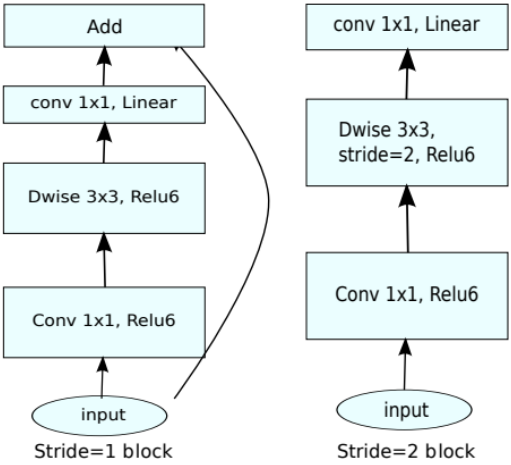
4) Softmax

Softmax adalah fungsi aktivasi yang umum digunakan untuk klasifikasi multi-kelas. Fungsi ini mengubah output menjadi probabilitas yang dijumlahkan menjadi 1, sehingga setiap kelas dapat diprediksi berdasarkan probabilitas. Berikut adalah persamaan fungsi aktivasi *softmax*:

$$f_j(z) = \frac{e^{zj}}{\sum_k e^{zk}} \quad (2.4)$$

7. MobileNetV2

MobileNetV2 merupakan salah satu arsitektur CNN yang dikembangkan dari MobileNet sebelumnya dan dirancang khusus untuk aplikasi mobile serta perangkat dengan sumber daya terbatas (Sastypratiwi et al., 2024). Arsitektur ini menggunakan *depthwise separable convolutions* untuk mengurangi jumlah parameter, serta *inverted residuals* dan *linear bottlenecks* untuk mempertahankan akurasi meskipun modelnya ringan (Sandler et al., 2018). MobileNetV2 memiliki ukuran yang lebih kecil dan efisiensi komputasi yang lebih baik dibandingkan arsitektur lain, tetapi tetap mampu memberikan performa yang baik dalam tugas pengenalan gambar.



Gambar 2. 10 Arsitektur MobileNetV2

8. *Transfer Learning*

Transfer learning adalah metode yang memanfaatkan model yang telah dilatih sebelumnya dengan menggunakan dataset besar untuk diterapkan kembali pada tugas yang berbeda. Metode ini menggunakan pengetahuan yang diperoleh dari tugas yang telah dipelajari untuk digunakan pada tugas yang lain. Tujuan penggunaan metode ini adalah untuk mengatasi masalah yang sering terjadi ketika menggunakan dataset kecil dalam *deep learning*. Untuk menggunakan metode ini, terdapat tiga cara, yakni *fixed feature extractor*, *fine-tuning*, dan *combined method* (Putra et al., 2021).

9. *Confusion Matrix*

Confusion matrix adalah matriks yang digunakan untuk mengevaluasi kinerja model klasifikasi dengan menunjukkan jumlah klasifikasi yang benar dan salah untuk setiap kelas. Setiap kolom mewakili kelas yang diprediksi, sementara setiap baris mewakili kelas yang sebenarnya (Markoulidakis et al., 2021). Pada klasifikasi multi-kelas, *confusion matrix* akan memiliki lebih dari dua baris dan kolom, masing-masing mewakili kelas yang berbeda. Gambar di bawah ini menunjukkan contoh dari *confusion matrix* pada klasifikasi multi-kelas.

		Predicted Class			
		C_1	**C_2**	**...**	**C_N**
Actual Class	**C_1**	$C_{1,1}$	FP	...	$C_{1,N}$
	C_2	FN	TP	...	FN
	...	...	...	...	...
	C_N	$C_{N,1}$	FP	...	$C_{N,N}$

Gambar 2. 11 Confusion Matrix

Pada klasifikasi multi-kelas, matriks berukuran NxN, di mana N adalah jumlah kelas yang ada, seperti C_1, C_2, ..., C_n. Matriks ini terdiri dari beberapa elemen utama seperti *true positive* (TP), *false positive* (FP), dan *false negative* (FN). Berdasarkan elemen-elemen

tersebut, metrik evaluasi seperti *accuracy*, *precision*, *recall*, dan *F1 score* dapat dihitung untuk mengukur kinerja model klasifikasi. Berikut adalah penjelasannya:

a. Accuracy

Accuracy merupakan rasio prediksi benar (positif dan negatif) dengan keseluruhan data. *Accuracy* dapat dihitung dengan rumus:

$$Accuracy = \frac{\sum_{i=1}^{n} TP(C_i)}{N} \qquad (2.5)$$

Keterangan:

i = 1, 2, 3, ...

n = Jumlah seluruh kelas

C_i = Kelas ke-1, 2, 3, ..., n

N = Jumlah seluruh data

TP = Jumlah data yang benar milik kelas tersebut dan diprediksi benar.

b. Precision

Precision merupakan rasio prediksi benar dibandingkan dengan seluruh prediksi yang diberikan untuk kelas tersebut. *Precision* dapat dihitung dengan rumus:

$$Precision(C_i) = \frac{TP(C_i)}{TP(C_i)+FP(C_i)} \qquad (2.6)$$

Keterangan:

TP = Jumlah data yang benar milik kelas tersebut dan diprediksi benar.

FP = Jumlah data yang diprediksi sebagai kelas tersebut, tetapi sebenarnya bukan.

c. Recall

Recall merupakan rasio prediksi benar dibandingkan dengan seluruh data yang sebenarnya milik kelas tersebut. R*ecall* dapat dihitung dengan rumus:

$$Recall(C_i) = \frac{TP(C_i)}{TP(C_i)+FN(C_i)} \qquad (2.7)$$

Keterangan:

TP = Jumlah data yang benar milik kelas tersebut dan diprediksi benar.

FN = Jumlah data yang sebenarnya milik kelas tersebut, tetapi diprediksi sebagai kelas lain.

d. F1 Score

F1 score merupakan rata-rata dari *precision* dan *recall*. *F1 score* dapat dihitung dengan rumus:

$$F1\ score(C_i) = 2 * \frac{Precision(C_i)*Recall(C_i)}{Precision(C_i)+Recall(C_i)} \qquad (2.8)$$

10. *Tools and Library*

a. Tools

1) Android Studio

Android Studio adalah *Integrated Development Environment* (IDE) resmi yang dikembangkan oleh Google untuk pengembangan aplikasi Android. IDE ini dibangun di atas platform IntelliJ IDEA dari JetBrains dan dirancang khusus untuk mempermudah pengembangan aplikasi Android dengan menyediakan berbagai fitur canggih. Android Studio mendukung bahasa pemrograman Java dan Kotlin, serta menawarkan berbagai alat yang membantu dalam proses pengembangan, seperti editor kode yang canggih, alat debugging, serta emulator Android yang memungkinkan pengujian aplikasi secara langsung pada berbagai perangkat virtual.

2) Google Colab

Google Colab (*Collaboratory*) adalah layanan *cloud* berbasis Jupyter Notebook yang dikembangkan oleh Google, yang memungkinkan pengguna untuk menulis dan

menjalankan kode Python secara langsung di browser tanpa perlu mengonfigurasi lingkungan pengembangan secara lokal. Colab menyediakan keunggulan seperti akses gratis ke GPU dan TPU yang berguna untuk menjalankan komputasi berat, seperti pelatihan model *machine learning*. Selain itu, Colab juga mendukung kolaborasi *real-time*, memungkinkan banyak pengguna untuk bekerja bersama dalam satu proyek secara bersamaan.

3) Kotlin

Kotlin adalah bahasa pemrograman yang dikembangkan oleh JetBrains pada tahun 2010, terinspirasi oleh bahasa pemrograman seperti Java, Scala, JavaScript, C#, dan Groovy. Kotlin mendukung pengembangan aplikasi untuk berbagai platform, baik untuk server atau backend, website, maupun aplikasi mobile Android, serta menawarkan kemampuan multiplatform dan mendukung berbagai paradigma pemrograman.

4) Python

Python adalah bahasa pemrograman multifungsi yang dirilis pada tahun 1991 oleh Guido van Rossum (GvR). Python menjadi bahasa yang sangat populer di kalangan data scientist dan praktisi *machine learning* berkat ekosistemnya yang kaya dengan *library* serta alat yang memudahkan proses analisis data, visualisasi, hingga pengembangan model *machine learning*.

b. Library

1) Numpy

Numpy merupakan *library* Python yang digunakan untuk komputasi ilmiah. *Library* ini memiliki kemampuan untuk membuat array multidimensi dan juga digunakan untuk berbagai operasi matematis, seperti aljabar linear, serta operasi vektor dan matriks.

2) Pandas

Pandas merupakan *library* Python yang digunakan untuk manipulasi dan analisis data. *Library* ini memiliki dua jenis struktur data yaitu Series dan DataFrame. Series merupakan struktur data satu dimensi berbentuk array,

sedangkan DataFrame merupakan struktur data dua dimensi yang menyerupai tabel, dengan baris dan kolom.

3) *Matplotlib*

Matplotlib merupakan *library* Python yang digunakan untuk membuat plot atau visualisasi data dalam bentuk dua dimensi. *Library* ini mendukung pembuatan berbagai jenis plot, seperti histogram, scatter plot, grafik batang, dan pie chart.

4) *Scikit-learn*

Scikit-learn merupakan *library* Python yang digunakan untuk pembelajaran mesin. *Library* ini menyediakan berbagai alat dan algoritma yang mempermudah pengembangan model *machine learning* untuk berbagai tugas seperti klasifikasi, regresi, dan lain-lain.

5) *Tensorflow*

Tensorflow merupakan *library* Python yang digunakan untuk membangun dan melatih model *machine learning*. *Library* ini dapat diterapkan dalam berbagai bidang, seperti *deep learning*, *computer vision*, dan *Natural Language Processing* (NLP).

B. Kajian Penelitian yang Relevan

Tabel 2. 1 Kajian Penelitian yang Relevan

No	Penelitian	Hasil Penelitian
1	The Hybrid Features and Supervised Learning for Batik Pattern Classification (Winarno et al., 2023)	Penelitian ini mengklasifikasikan motif batik Semarang menggunakan ANN, KNN, Decision Tree, Naïve Bayes, dan SVM. Dataset terdiri dari 3.000 gambar dari 10 kelas motif batik, dengan fitur yang diekstraksi dari warna, bentuk, dan tekstur. Hasil penelitian menunjukkan bahwa ANN memberikan akurasi tertinggi sebesar 99,76%, dengan fitur warna sebagai faktor paling dominan dalam klasifikasi.
2	Classification of Papuan Batik Motifs Using Deep Learning and Data Augmentation (Aras et al., 2022)	Penelitian ini mengklasifikasikan motif batik Papua menggunakan CNN dengan model VGG-16 dan ResNet-50. Dataset terdiri dari 213 gambar dari 4 kelas motif batik, dengan augmentasi data untuk meningkatkan akurasi model. Hasil penelitian menunjukkan bahwa VGG-16 dan ResNet-50 awalnya mencapai akurasi 78,79% dan 81,82%, kemudian setelah

		augmentasi, akurasi meningkat menjadi 84,85% dan 87,88%.
3	Classification of Batik Authenticity Using Convolutional Neural Network Algorithm with Transfer Learning Method (Putra et al., 2021)	Penelitian ini mengklasifikasikan jenis batik (batik tulis dan batik cetak) menggunakan MobileNetV2 dengan *transfer learning*. Dataset terdiri dari 566 gambar batik, dengan augmentasi data untuk meningkatkan akurasi model. Hasil penelitian menunjukkan bahwa MobileNetV2 mencapai akurasi validasi 94,74%, dengan akurasi batik tulis 87,09% dan batik cetak 88%.
4	ResNet-50 for Classifying Indonesian Batik with Data Augmentation (Negara et al., 2021)	Penelitian ini mengklasifikasikan motif batik menggunakan CNN dengan model ResNet-50 serta menerapkan augmentasi data. Dataset terdiri dari 300 gambar dari 50 kelas motif batik, yang bertambah menjadi 1.500 gambar setelah augmentasi. Hasil penelitian menunjukkan bahwa ResNet-50 awalnya mencapai akurasi 95%, kemudian setelah augmentasi, akurasi meningkat menjadi 100%.

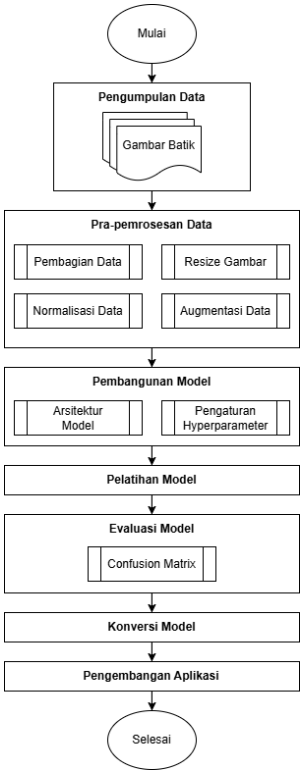
5	Implementation of Data Augmentation Using Convolutional Neural Network for Batik Classification (Uswatun Khasanah et al., 2020)	Penelitian ini mengklasifikasikan motif batik menggunakan VGG-16 dengan *fine-tuning*, serta menerapkan 8 teknik augmentasi data. Dataset terdiri dari 500 gambar dari 5 kelas motif batik. Hasil penelitian menunjukkan bahwa VGG-16 awalnya mencapai akurasi 95,83%, kemudian setelah augmentasi, akurasi meningkat menjadi 98,96%.

Dari Tabel 2.1, diketahui bahwa klasifikasi batik telah banyak diteliti sebelumnya. Pada penelitian ini, penulis akan mengembangkan sistem klasifikasi motif batik Semarang menggunakan MobileNetV2 dengan *transfer learning* serta menerapkan augmentasi data yang direkomendasikan dari penelitian sebelumnya untuk meningkatkan akurasi model.

Batik Semarang dipilih karena belum ada penelitian yang mengklasifikasikannya menggunakan *Convolutional Neural Network*. Sementara itu, MobileNetV2 digunakan karena arsitekturnya yang ringan dan efisien, sehingga dapat diterapkan pada perangkat Android, yang masih jarang dikembangkan dalam penelitian klasifikasi batik.

BAB III
METODOLOGI PENELITIAN

Metodologi penelitian merupakan pedoman atau tahapan sistematis yang digunakan dalam penelitian ini. Metodologi ini mencakup serangkaian tahapan yang digunakan untuk mengembangkan model klasifikasi batik Semarang serta mengintegrasikannya ke dalam aplikasi. Rangkaian tahapan tersebut ditunjukkan pada Gambar 3.1.



Gambar 3. 1 Metodologi Penelitian

A. Pengumpulan Data

Pada penelitian ini, penulis menggunakan data sekunder, yaitu data yang diperoleh secara tidak langsung atau melalui sumber yang telah tersedia sebelum penelitian dilakukan. Dataset yang digunakan adalah Semarang Batik Dataset, yang sebelumnya telah digunakan dalam penelitian (Winarno et al., 2023) dan kini tersedia di repository Kaggle.

B. Pra-pemrosesan Data

Pra-pemrosesan adalah langkah yang bertujuan untuk meningkatkan kualitas data, menghilangkan noise yang mungkin ada, dan menentukan bagian data yang akan digunakan pada langkah-langkah berikutnya (Wona et al., 2023). Data batik diolah dan dipersiapkan sebelum dimasukkan ke dalam model. Pra-pemrosesan data meliputi:

1. Pembagian Data

   Pada penelitian ini, dataset dibagi menjadi tiga bagian, yaitu data latih, validasi, dan uji. Data latih digunakan untuk melatih model dalam mengenali motif batik Semarang, sedangkan data validasi digunakan untuk memantau serta mengevaluasi kinerja model selama proses pelatihan. Setelah pelatihan selesai, data uji digunakan untuk menguji

kinerja model. Pembagian data dilakukan dengan rasio 80% untuk data latih dan 20% untuk data uji, di mana dari data latih tersebut, 20% digunakan untuk data validasi.

2. *Resize* Gambar

   Resize gambar merupakan teknik yang digunakan untuk mengubah ukuran gambar, baik dengan memperbesar maupun memperkecilnya. Dalam penelitian ini, ukuran gambar disesuaikan menjadi 224x224 piksel agar semua gambar dalam dataset memiliki dimensi yang sama. Penyesuaian ukuran gambar ini penting agar model dapat memproses data secara konsisten dan optimal.

3. Normalisasi Gambar

   Normalisasi gambar adalah teknik yang digunakan untuk mengubah nilai piksel gambar ke dalam rentang tertentu, biasanya antara 0 hingga 1. Dengan menyesuaikan skala gambar, diharapkan model tidak terfokus pada fitur-fitur tertentu yang mungkin memiliki nilai piksel yang lebih besar. Proses ini dapat meningkatkan kemampuan model dalam belajar dan mempercepat konvergensi selama sesi pelatihan.

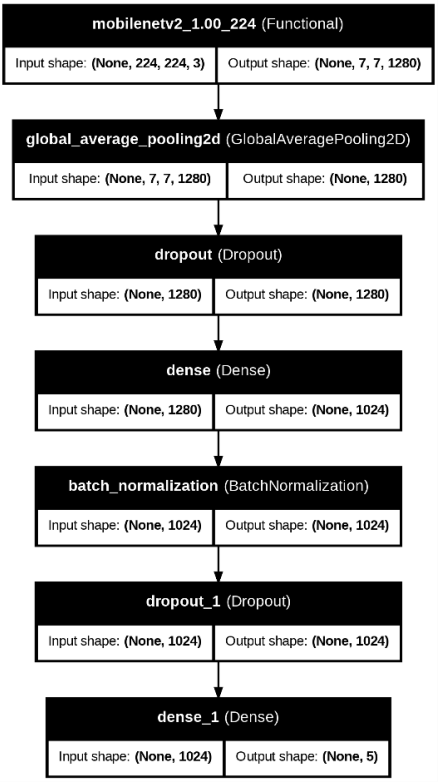
4. Augmentasi Gambar

Augmentasi gambar adalah teknik yang digunakan untuk memperbanyak jumlah gambar pelatihan dengan memodifikasi gambar asli guna menciptakan variasi gambar yang lebih beragam (Uswatun Khasanah et al., 2020). Proses augmentasi tidak menambah gambar secara fisik atau permanen, melainkan menghasilkan gambar baru secara sintetis yang bersifat sementara dan disimpan di dalam memori (Akhiar, 2021).

Dalam penelitian ini, teknik augmentasi gambar yang dilakukan berupa rotasi, *zoom*, pergeseran sudut (*shear*), dan pencerminan gambar secara horizontal. Dengan melakukan augmentasi, masalah *overfitting* yang sering muncul selama pelatihan model dapat berkurang (Negara et al., 2021).

C. Pembangunan Model

1. Arsitektur Model



Gambar 3. 2 Arsitektur Model

Pada penelitian ini, digunakan arsitektur MobileNetV2 sebagai *feature extractor* untuk klasifikasi gambar batik. MobileNetV2 dipilih karena memiliki struktur yang ringan dan efisien dalam ekstraksi fitur, sehingga cocok untuk diterapkan pada

perangkat dengan keterbatasan sumber daya komputasi.

Berikut ini penjelasan lebih rinci terkait arsitektur model yang dibangun untuk tahapan klasifikasi batik, seperti yang ditunjukkan pada Gambar 3.2.

a. Model menerima input berupa gambar dengan ukuran (224, 224, 3). Ukuran ini menunjukkan bahwa gambar memiliki lebar 224 piksel, tinggi 224 piksel, dan memiliki 3 channel warna (RGB).
b. Gambar diproses menggunakan MobileNetV2 yang telah dilatih sebelumnya pada dataset ImageNet. Model ini digunakan tanpa *fully connected layer* di bagian akhir, sehingga hanya berfungsi sebagai *feature extractor*. Output dari MobileNetV2 berupa *feature map* berukuran 7 × 7 × 1280.
c. Dilakukan operasi *Global Average Pooling* yang mengubah *feature map* berukuran 7 × 7 × 1280 menjadi vektor berukuran (1, 1280). Operasi ini bertujuan untuk mereduksi dimensi fitur tanpa kehilangan informasi penting.
d. Vektor fitur hasil ekstraksi diproses oleh *dense layer* dengan 1024 neuron, yang berfungsi untuk menangkap hubungan kompleks antara fitur yang

telah diekstraksi. Fungsi aktivasi yang digunakan pada layer ini adalah ReLU.

e. *Batch Normalization* diterapkan setelah *dense layer* untuk menormalkan distribusi aktivasi, sehingga mempercepat konvergensi selama pelatihan.
f. Untuk mencegah *overfitting*, diterapkan *Dropout layer* dengan rate tertentu, yang berfungsi untuk menonaktifkan sebagian neuron secara acak selama pelatihan.
g. Pada tahap akhir, diterapkan *dense layer* dengan 5 neuron, sesuai dengan jumlah kelas batik yang diklasifikasikan. Fungsi aktivasi yang digunakan adalah softmax, yang mengubah output menjadi probabilitas untuk setiap kelas.

2. Konfigurasi Hyperparameter

Hyperparameter merupakan variabel yang ditentukan sebelum proses pelatihan model dan memiliki peran penting dalam memengaruhi performa serta akurasi model. Konfigurasi hyperparameter yang tepat dapat membantu model mencapai konvergensi lebih cepat dan terhindar dari *overfitting*. Beberapa hyperparameter yang umum digunakan dalam arsitektur CNN antara lain *epoch*, *batch size*, *optimizer,* dan *learning rate* (Rochmawati et al., 2021).

Pada penelitian ini, dilakukan percobaan dengan berbagai nilai hyperparameter untuk menemukan konfigurasi yang menghasilkan pelatihan model yang optimal. Tabel 3.1 berikut menampilkan nilai-nilai hyperparameter yang digunakan dalam percobaan.

Tabel 3. 1 Nilai-nilai Hyperparameter

Hyperparameter	**Nilai yang Dicoba**
Epoch	10, 20, 30
Batch Size	32, 64, 128
Optimizer	Adam
Learning Rate	0,001, 0.0001, 0,00001

Setelah dilakukan beberapa percobaan, berikut adalah pertimbangan dalam pemilihan nilai akhir dari masing-masing hyperparameter:

a. *Epoch*: Percobaan awal menunjukkan bahwa 10 *epoch* belum cukup untuk menghasilkan akurasi yang stabil, sementara 30 *epoch* cenderung menyebabkan *overfitting*. Oleh karena itu, digunakan 20 epoch sebagai jumlah yang optimal untuk mencapai keseimbangan antara akurasi dan waktu pelatihan.
b. *Batch Size*: Percobaan dengan *batch size* 32 menghasilkan pelatihan yang stabil namun

memakan waktu lebih lama, sedangkan *batch size* 128 mempercepat proses pelatihan namun menurunkan akurasi model. *Batch size* 64 dipilih karena memberikan hasil terbaik dalam hal efisiensi dan performa.

c. *Optimizer*: *Optimizer* Adam dipilih karena kemampuannya dalam menyesuaikan *learning rate* secara adaptif sehingga mempercepat proses konvergensi dan meningkatkan kestabilan pelatihan dibandingkan *optimizer* lainnya.
d. *Learning Rate*: Percobaan menunjukkan bahwa *learning rate* 0.001 menyebabkan fluktuasi nilai *loss* yang besar dan kurang stabil, sementara *learning rate* 0.00001 membuat proses pelatihan menjadi sangat lambat. *Learning rate* 0.0001 dipilih sebagai nilai yang memberikan kestabilan dan kecepatan pelatihan yang seimbang.

Dengan mempertimbangkan hasil percobaan tersebut, konfigurasi akhir yang digunakan dalam penelitian ini adalah ***epoch* sebanyak 20, *batch size* sebesar 64, *optimizer* Adam, dan *learning rate* sebesar 0.0001**.

D. Pelatihan Model

Pada tahap ini, dataset hasil pra-pemrosesan digunakan untuk melatih model agar dapat mengenali fitur penting pada gambar batik. Model dalam penelitian ini menerapkan *transfer learning* dengan MobileNetV2 sebagai *base model* dan pengekstraksi fitur, kemudian dilengkapi dengan beberapa lapisan tambahan untuk menyesuaikan tugas klasifikasi.

Pelatihan dilakukan dengan menggunakan hyperparameter yang telah ditetapkan sebelumnya, seperti jumlah *epoch*, *batch size*, *optimizer*, dan *learning rate*. Konfigurasi ini bertujuan agar model dapat mempelajari pola data dengan baik, sehingga menghasilkan performa yang konsisten pada data validasi baik dari segi akurasi maupun *loss* selama proses pelatihan. Selama pelatihan, grafik akurasi dan loss pada data latih serta data validasi dipantau di setiap *epoch* untuk memastikan model belajar secara optimal dan menghindari *overfitting*.

Selain itu, diterapkan *callback* seperti *EarlyStopping* untuk menghentikan pelatihan lebih awal apabila performa model tidak mengalami peningkatan setelah beberapa *epoch*, serta *ModelCheckpoint* untuk menyimpan bobot model terbaik selama pelatihan berlangsung.

E. Evaluasi Model

Evaluasi model merupakan tahapan penting dalam penelitian ini, di mana model yang telah dilatih akan diuji menggunakan data uji untuk mengukur efektivitasnya. Evaluasi dilakukan dengan menggunakan *confusion matrix*, yang memberikan informasi mendetail tentang hasil prediksi model dan menjadi dasar untuk menghitung metrik seperti *accuracy*, *precision*, *recall*, dan *F1 score*. Perhitungan keempat metrik tersebut dilakukan berdasarkan nilai-nilai yang diperoleh dari *confusion matrix*, menggunakan rumus (2.5) untuk *accuracy*, (2.6) untuk *precision*, (2.7) untuk *recall*, dan (2.8) untuk *F1 score*.

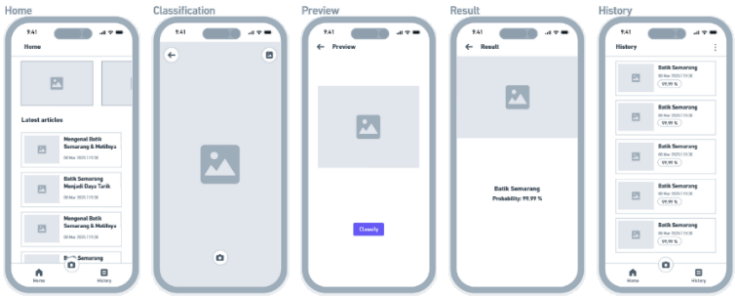
F. Konversi Model

Konversi model merupakan tahapan lanjutan setelah tahap pelatihan dan evaluasi selesai dilakukan. Pada tahap ini, model yang telah dilatih dengan format Keras (.h5) dikonversi ke format *TensorFlow Lite* (.tflite) agar dapat digunakan pada perangkat Android. Model TFLite yang telah dihasilkan kemudian akan dibundle ke dalam proyek aplikasi Android dan digunakan sebagai model utama untuk melakukan klasifikasi gambar batik. Dengan pendekatan ini, proses inferensi dapat dilakukan secara langsung di perangkat pengguna.

G. Aplikasi

Aplikasi klasifikasi batik Semarang digunakan sebagai media bantu dalam penelitian ini. Aplikasi ini memungkinkan pengguna mengunggah atau mengambil gambar batik untuk kemudian diklasifikasikan menggunakan model *machine learning* yang telah ditanamkan di dalam aplikasi. Proses klasifikasi dilakukan langsung di perangkat (*on-device*), sehingga aplikasi tetap dapat digunakan tanpa memerlukan koneksi internet.

Untuk merancang tampilan aplikasi dengan baik, dibuatlah wireframe sebagai representasi visual dari struktur dan tata letak elemen pada setiap halaman. Wireframe ini berfungsi sebagai sketsa awal antarmuka pengguna yang memudahkan pemahaman alur dan fungsi aplikasi sebelum pengembangan lebih lanjut (Imanuel & Tobing, 2023). Berikut ini adalah wireframe dari halaman-halaman utama aplikasi:



Gambar 3. 3 *Wireframe* Aplikasi

BAB IV
HASIL DAN PEMBAHASAN

A. Hasil Pengumpulan Data

Pada penelitian ini, dataset yang digunakan berasal dari Semarang Batik Dataset, yang tersedia di *repository* Kaggle. Dataset ini berisi 10 motif batik khas Semarang, namun dalam penelitian ini hanya digunakan 5 motif batik, yaitu Asam Arang, Blekok Srondol, Lawang Sewu, Tugu Muda, dan Warak Ngendog. Dataset batik untuk kelima motif tersebut ditampilkan pada Lampiran 3.

Dataset tersebut disimpan dalam format ZIP di Google Drive, sehingga proses pemuatan ke dalam Google Colab dilakukan dengan mounting Google Drive dan mengekstrak file ZIP ke dalam direktori Google Colab. Berikut adalah kode yang digunakan untuk proses tersebut:

```
from google.colab import drive
import zipfile
import os

# Mount Google Drive
drive.mount('/content/drive')

# Path ke zip file
zip_path = '/content/drive/MyDrive/Dataset/semarang-batik.zip'

# Path tujuan unzip
extract_path = '/content/semarang-batik-dataset'

# Unzip file
with zipfile.ZipFile(zip_path, 'r') as zip_ref:
    zip_ref.extractall(extract_path)

print(f"Dataset berhasil di-unzip ke {extract_path}")
```

Gambar 4. 1 Kode Pemuatan Dataset

Kode pada Gambar 4.1 merupakan kode yang digunakan untuk mengekstrak dataset gambar batik dari file arsip berformat ZIP yang tersimpan di Google Drive ke dalam direktori kerja Google Colab. Pada bagian awal, dilakukan impor beberapa pustaka yang diperlukan, yaitu google.colab.drive untuk mengakses Google Drive, zipfile untuk menangani file ZIP, dan os untuk interaksi dengan sistem file serta pengelolaan path direktori.

Langkah selanjutnya adalah melakukan mounting Google Drive ke lingkungan Google Colab dengan perintah drive.mount('/content/drive'), sehingga file dalam Google Drive dapat diakses secara langsung dari Colab. Setelah proses mounting berhasil, path menuju file ZIP dataset disimpan dalam variabel zip_path, sementara direktori tujuan ekstraksi disimpan dalam variabel extract_path.

Proses ekstraksi dilakukan dengan membuka file ZIP menggunakan zipfile.ZipFile() dalam mode baca ("r"), kemudian seluruh isi file ZIP diekstrak ke folder tujuan menggunakan fungsi .extractall(). Setelah proses ekstraksi selesai, ditampilkan pesan konfirmasi melalui perintah print() yang menunjukkan bahwa dataset telah berhasil diekstrak ke lokasi yang ditentukan.

Setelah proses ekstraksi selesai dan dataset berhasil disiapkan di direktori kerja, langkah selanjutnya adalah

melakukan visualisasi jumlah gambar pada setiap kelas batik guna memastikan distribusinya seimbang. Kode untuk proses ini ditunjukkan pada Gambar 4.2 berikut.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

main_path = "/content/semarang-batik-dataset"

file_name = []
labels = []
full_path = []

for path, subdirs, files in os.walk(main_path):
    for name in files:
        full_path.append(os.path.join(path, name))
        labels.append(path.split('/')[-1])
        file_name.append(name)

df = pd.DataFrame({"path":full_path,'file_name':file_name,"labels":labels})

# Plot distribusi gambar di setiap kelas
Label = df['labels']
plt.figure(figsize = (6,6))
sns.set_style("darkgrid")
plot_data = sns.countplot(Label)
```

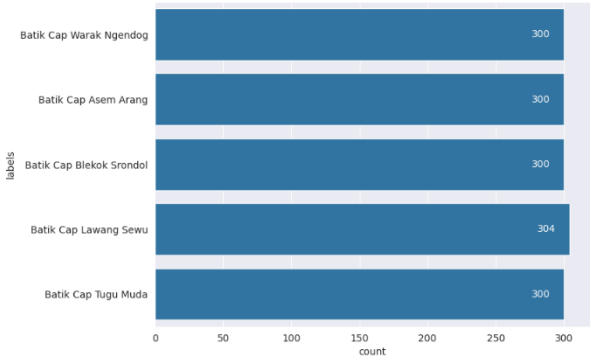
Gambar 4. 2 Kode Visualisasi Jumlah Gambar Tiap Kelas

Kode pada Gambar 4.2 merupakan kode yang digunakan untuk memvisualisasikan distribusi jumlah gambar pada setiap kelas batik dalam bentuk grafik. Pada bagian awal, dilakukan impor beberapa pustaka yang diperlukan, yaitu *pandas* untuk pengolahan data, *matplotlib.pyplot* dan *seaborn* untuk visualisasi grafik. Variabel main_path digunakan untuk menyimpan direktori utama tempat dataset disimpan.

Selanjutnya, proses iterasi dilakukan terhadap seluruh file dalam struktur folder menggunakan os.walk(). Setiap nama file dan nama folder (yang merepresentasikan

label kelas) disimpan ke dalam dua list terpisah, yaitu file_name dan labels. Informasi lengkap mengenai path file, nama file, dan label kelas kemudian dimasukkan ke dalam sebuah objek DataFrame bernama df.

Pada bagian akhir, visualisasi data dibuat menggunakan fungsi countplot() dari pustaka Seaborn dengan orientasi horizontal (menggunakan parameter y='labels'). Visualisasi ini bertujuan untuk mengetahui jumlah gambar pada setiap kelas batik sekaligus memastikan bahwa distribusinya seimbang sebelum proses pelatihan model dilakukan. Hasil visualisasi tersebut ditampilkan pada Gambar 4.3 berikut.



Gambar 4. 3 Distribusi Jumlah Gambar Tiap Kelas

Berdasarkan Gambar 4.3, dapat dilihat bahwa jumlah gambar dalam setiap kelas hampir merata, dengan sebagian besar kelas memiliki 300 gambar, sementara satu kelas memiliki 304 gambar. Perbedaan jumlah ini relatif

kecil dan tidak berdampak signifikan terhadap keseimbangan dataset secara keseluruhan. Dataset yang relatif seimbang sangat penting dalam pelatihan model klasifikasi, karena dapat mengurangi bias model terhadap kelas tertentu dan meningkatkan performa klasifikasi.

B. Hasil Pra-pemrosesan Data

Tahapan pra-pemrosesan yang dilakukan dalam penelitian ini terdiri dari pembagian data, normalisasi gambar, resize gambar, dan augmentasi gambar.

1. Pembagian Data

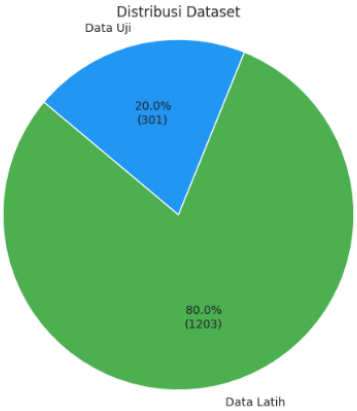
   Pada penelitian ini, dataset dibagi menjadi dua bagian: data latih dan data uji. Pembagian data ini dilakukan menggunakan fungsi train_test_split dari pustaka sklearn.model_selection. Pembagian data ini bertujuan untuk memisahkan data yang digunakan untuk melatih model dan untuk menguji kinerjanya. Kode pembagian dataset ditunjukkan pada Gambar 4.4 berikut.

```
from sklearn.model_selection import train_test_split

# Memisahkan dataset menjadi path gambar (X) dan label kelas (y)
X = df['path']
y = df['labels']

# Membagi data menjadi 80% data latih dan 20% data uji
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=123)
```

Gambar 4. 4 Kode Pembagian Data

Kode pada Gambar 4.4 digunakan untuk membagi dataset menjadi data latih dan data uji. Variabel X berisi path gambar dari kolom 'path', sedangkan y berisi label dari kolom 'labels' pada DataFrame df. Fungsi train_test_split digunakan dengan parameter test_size=0.2, artinya 20% data digunakan sebagai data uji dan 80% sisanya sebagai data latih. Parameter random_state=123 memastikan pembagian data yang konsisten setiap kali kode dijalankan. Hasil pembagian disimpan dalam empat variabel: X_train, X_test, y_train, dan y_test.

Untuk memberikan gambaran mengenai proporsi hasil pembagian data, Gambar 4.5 berikut menyajikan visualisasi dalam bentuk diagram pie yang menunjukkan distribusi antara data latih dan data uji.



Gambar 4. 5 Hasil Pembagian Data

2. *Resize* Gambar

Setelah proses pembagian dataset selesai, tahap selanjutnya adalah *resize* gambar. *Resize* dilakukan agar seluruh gambar memiliki ukuran yang seragam sebelum digunakan dalam pelatihan model. Dalam penelitian ini, ukuran yang digunakan adalah 224×224 piksel, menyesuaikan dengan konfigurasi input pada arsitektur MobileNetV2 yang digunakan.

Proses *resize* gambar ini dilakukan menggunakan ImageDataGenerator dari *TensorFlow*, sebagaimana ditunjukkan pada Gambar 4.6 berikut.

```
train_data = data_gen.flow_from_directory (
    TRAIN_DIR,
    target_size=(224, 224)
)
validation_data = data_gen.flow_from_directory (
    TRAIN_DIR,
    target_size=(224, 224)
)
test_data = test_data_gen.flow_from_directory (
    TEST_DIR,
    target_size=(224, 224)
)
```

Gambar 4. 6 Kode Resize Gambar

Kode pada Gambar 4.6 menunjukkan penggunaan metode flow_from_directory() dengan parameter target_size=(224, 224) untuk memuat dan mengubah ukuran gambar dari direktori dataset. Proses ini diterapkan pada data latih (*train_data*), data validasi (*validation_data*), dan data uji (*test_data*).

Untuk memberikan gambaran mengenai hasil *resize* gambar, Lampiran 4 menyajikan contoh gambar sebelum dan setelah diubah ukurannya menjadi 224×224 piksel.

3. Normalisasi Gambar

Setelah proses *resize* gambar selesai, tahap selanjutnya adalah normalisasi gambar. Normalisasi dilakukan dengan mengubah nilai piksel gambar yang semula berada pada rentang [0, 255] menjadi [0, 1]. Tujuannya adalah untuk meningkatkan efisiensi dan stabilitas pelatihan, karena CNN cenderung lebih cepat beradaptasi dengan data yang memiliki skala kecil.

Proses normalisasi gambar ini dilakukan menggunakan ImageDataGenerator dari *TensorFlow* dengan parameter rescale=1./255, sebagaimana ditunjukkan pada Gambar 4.7 berikut.

```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator

data_gen = ImageDataGenerator(rescale=1./255)
test_data_gen = ImageDataGenerator(rescale=1./255)
```

Gambar 4. 7 Kode Normalisasi Gambar

Kode pada Gambar 4.7 menunjukkan inisialisasi dua objek ImageDataGenerator, yaitu data_gen untuk data latih dan test_data_gen untuk data uji. Keduanya menggunakan parameter rescale=1./255, yang

berfungsi untuk membagi setiap nilai piksel gambar dengan angka 255 sehingga nilainya berada pada rentang [0, 1].

Untuk memberikan gambaran mengenai hasil normalisasi gambar, Lampiran 5 menyajikan perbandingan nilai piksel pada gambar sebelum dan setelah dinormalisasi.

4. Augmentasi Gambar

Setelah proses normalisasi gambar selesai, tahap selanjutnya adalah augmentasi gambar. Augmentasi dilakukan untuk meningkatkan variasi dalam dataset tanpa harus menambah data baru secara manual. Dalam penelitian ini, augmentasi diterapkan pada data latih menggunakan beberapa teknik, seperti rotasi, *zoom*, *shear*, dan pencerminan horizontal.

Proses augmentasi ini dilakukan menggunakan ImageDataGenerator dari *TensorFlow*, sebagaimana ditunjukkan pada Gambar 4.8 berikut.

```
data_gen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    zoom_range=0.2,
    shear_range=0.2,
    horizontal_flip=True,
    fill_mode = 'nearest',
    validation_split=0.2
)
```

Gambar 4. 8 Kode Augmentasi Gambar

Kode pada Gambar 4.8 menunjukkan proses augmentasi gambar pada data latih menggunakan ImageDataGenerator dari *TensorFlow*. Beberapa parameter yang digunakan antara lain rotation_range=20 untuk merotasi gambar hingga 20 derajat, zoom_range=0.2 untuk memperbesar atau memperkecil gambar hingga 20%, serta shear_range=0.2 untuk melakukan pergeseran sudut gambar hingga 20%. Parameter horizontal_flip=True digunakan untuk membalik gambar secara horizontal, sedangkan fill_mode='nearest' digunakan untuk mengisi piksel kosong yang muncul akibat transformasi dengan nilai piksel terdekat. Selain itu, proses normalisasi diterapkan melalui rescale=1./255, dan validation_split=0.2 digunakan untuk memisahkan 20% data latih sebagai data validasi.

Seluruh transformasi augmentasi diterapkan secara acak pada setiap batch data selama proses pelatihan berlangsung, sehingga model menerima variasi gambar yang berbeda pada setiap *epoch*. Untuk memberikan gambaran mengenai hasil augmentasi, Lampiran 6 menyajikan gambar asli beserta variasi hasil augmentasinya.

C. Hasil Pembangunan Model

1. Arsitektur Model

Dalam penelitian ini, digunakan *transfer learning* dengan arsitektur MobileNetV2 sebagai *feature extractor* untuk proses klasifikasi batik. *Transfer Learning* adalah teknik pembelajaran mesin di mana model yang telah dilatih sebelumnya pada dataset yang besar, seperti ImageNet, digunakan kembali untuk tugas baru dengan sedikit penyesuaian. Teknik ini memungkinkan model untuk memanfaatkan fitur yang telah dipelajari sebelumnya, sehingga dapat meningkatkan performa dan mempercepat proses pelatihan tanpa harus melatih model dari awal.

Sementara itu, MobileNetV2 dipilih karena memiliki struktur yang ringan serta efisien dalam mengekstraksi fitur dari gambar, sehingga cocok untuk diterapkan pada perangkat dengan keterbatasan sumber daya komputasi seperti Android. Model ini dimuat dengan bobot hasil pelatihan pada dataset ImageNet, dengan menghilangkan lapisan klasifikasi atas (*include_top=False*) agar dapat disesuaikan dengan tugas klasifikasi batik. Ukuran input yang digunakan adalah 224×224 piksel dengan 3 channel

warna (RGB). Implementasi pemuatan MobileNetV2 ditunjukkan dalam kode berikut:

```
base_model = MobileNetV2 (
    include_top=False,
    weights="imagenet",
    input_shape=(224, 224, 3)
)
```

Gambar 4. 9 Kode Pemuatan MobileNetV2

Dalam penerapan *Transfer Learning*, lapisan-lapisan MobileNetV2 yang telah dilatih sebelumnya tidak dilatih ulang untuk menjaga fitur-fitur yang telah dipelajari dari dataset ImageNet. Oleh karena itu, semua bobot pada model ini dibekukan (*trainable = False*), sehingga selama proses pelatihan hanya lapisan tambahan yang akan mengalami pembaruan parameter. Hal ini bertujuan untuk mencegah hilangnya informasi yang telah dipelajari oleh model dasar dan sekaligus mempercepat proses pelatihan. Proses pembekuan bobot dilakukan dengan kode berikut:

```
base_model.trainable = False
```

Gambar 4. 10 Kode Pembekuan Bobot MobileNetV2

Setelah MobileNetV2 dimuat sebagai *feature extractor*, model klasifikasi batik Semarang dibangun dengan menambahkan beberapa lapisan tambahan.

Lapisan pertama setelah MobileNetV2 adalah *GlobalAveragePooling2D*, yang berfungsi untuk mengubah fitur hasil ekstraksi menjadi vektor satu dimensi, sehingga jumlah parameter yang harus dilatih menjadi lebih sedikit dan lebih efisien dibandingkan penggunaan *Flatten*. Untuk mengurangi risiko *overfitting*, diterapkan *Dropout* dengan tingkat 50%, yang akan secara acak menonaktifkan sebagian neuron selama pelatihan.

Selanjutnya, model dilengkapi dengan lapisan *Dense* berisi 1.024 neuron dengan fungsi aktivasi ReLU, yang bertujuan untuk menangkap pola lebih kompleks dalam fitur yang telah diekstraksi. Selain itu, diterapkan *Batch Normalization* untuk menormalkan distribusi aktivasi guna meningkatkan stabilitas selama pelatihan. Kemudian, *Dropout* kembali diterapkan sebelum lapisan terakhir untuk semakin mengurangi risiko *overfitting*.

Lapisan terakhir dalam model ini adalah lapisan output, yang terdiri dari 5 neuron sesuai dengan jumlah kelas batik dalam dataset. Fungsi aktivasi *softmax* digunakan untuk menghasilkan probabilitas pada masing-masing kelas, sehingga model dapat menentukan kategori batik yang paling sesuai dengan

gambar input. Implementasi lengkap dari arsitektur model ini ditunjukkan pada kode berikut:

```
model = Sequential([
    base_model,
    GlobalAveragePooling2D(),
    Dropout(0.5),
    Dense(1024, activation='relu'),
    BatchNormalization(),
    Dropout(0.5),
    Dense(5, activation='softmax')
])
```

Gambar 4. 11 Kode Arsitektur Model

Setelah seluruh lapisan model tersusun, arsitektur model ditampilkan menggunakan fungsi *model.summary()*. Fungsi ini menampilkan informasi penting seperti jenis setiap lapisan, bentuk output dari masing-masing lapisan, serta jumlah parameter yang dapat dilatih dan yang dibekukan karena penggunaan model pralatih (*pretrained*) dalam pendekatan *transfer learning*. Ringkasan arsitektur model ditunjukkan pada Gambar 4.12 berikut.

Model: "sequential"

Layer (type)	Output Shape	Param #
mobilenetv2_1.00_224 (Functional)	(None, 7, 7, 1280)	2,257,984
global_average_pooling2d (GlobalAveragePooling2D)	(None, 1280)	0
dropout (Dropout)	(None, 1280)	0
dense (Dense)	(None, 1024)	1,311,744
batch_normalization (BatchNormalization)	(None, 1024)	4,096
dropout_1 (Dropout)	(None, 1024)	0
dense_1 (Dense)	(None, 5)	5,125

Total params: 3,578,949 (13.65 MB)
Trainable params: 1,318,917 (5.03 MB)
Non-trainable params: 2,260,032 (8.62 MB)

Gambar 4. 12 Hasil Arsitektur Model

2. Konfigurasi Hyperparameter dan *Callbacks*

Berdasarkan nilai-nilai hyperparameter yang telah ditentukan dalam Bab 3 (Tabel 3.1), tahap ini mengimplementasikan konfigurasi tersebut ke dalam model untuk memastikan proses pelatihan berjalan secara optimal. Hyperparameter yang digunakan meliputi jumlah *epoch* sebanyak 20, *batch size* sebesar 64, *optimizer* Adam, dan *learning rate* sebesar 0,0001. Konfigurasi ini diterapkan pada saat kompilasi model sebelum proses pelatihan, sebagaimana ditunjukkan pada Gambar 4.13 berikut.

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

Gambar 4. 13 Konfigurasi Hyperparameter

Pemilihan optimizer Adam dilakukan karena algoritma ini mampu menyesuaikan *learning rate* secara dinamis, sehingga proses pembelajaran dapat berlangsung lebih optimal. Kemudian, *loss function* yang digunakan adalah *categorical crossentropy*, karena penelitian ini merupakan klasifikasi multi-kelas. Sedangkan, metrik evaluasi yang digunakan adalah *accuracy*, karena untuk mengukur sejauh mana

model dapat mengklasifikasikan gambar batik dengan benar.

Selain itu, diterapkan mekanisme *callbacks* untuk meningkatkan efisiensi proses pelatihan dan menghindari permasalahan seperti *overfitting*. Dalam penelitian ini, digunakan dua jenis *callbacks*, yaitu *ModelCheckpoint* dan *EarlyStopping*. *ModelCheckpoint* berfungsi untuk menyimpan bobot model terbaik selama pelatihan berdasarkan akurasi validasi tertinggi. Sementara itu, *EarlyStopping* digunakan untuk menghentikan pelatihan lebih awal jika tidak terjadi peningkatan performa dalam beberapa *epoch* terakhir. Implementasi *callbacks* dalam penelitian ini ditunjukkan pada Gambar 4.14 berikut:

```
#Callbacks
checkpoint = ModelCheckpoint(
    'best_model.keras',
    monitor='val_accuracy',
    save_best_only=True,
    mode='max',
    verbose=1
)

early_stopping = EarlyStopping(
    monitor='val_loss',
    patience=3,
    mode='min',
    verbose=1
)
```

Gambar 4. 14 Implementasi *Callbacks*

D. Hasil Pelatihan Model

Pada tahap ini, model dilatih menggunakan dataset yang telah melalui tahap pra-pemrosesan agar mampu mengenali pola dan karakteristik unik dari setiap jenis batik. Pelatihan dilakukan dengan menyesuaikan sejumlah parameter yang telah ditentukan sebelumnya, seperti jumlah *epoch*, *batch size*, *optimizer*, dan *learning rate*. Proses pelatihan model ditunjukkan pada Gambar 4.15 berikut ini.

```
import time

start = time.time()

history=model.fit(
    x=train_data,
    batch_size=64,
    epochs=20,
    validation_data=validation_data,
    callbacks=[checkpoint, early_stopping]
)

stop = time.time()

duration = stop - start
minutes = int(duration // 60)
seconds = int(duration % 60)
print(f'Time = {minutes} minutes {seconds} seconds')
```

Gambar 4. 15 Kode Pelatihan Model

Gambar 4.15 menampilkan cuplikan kode pelatihan model menggunakan fungsi model.fit(). Pada proses ini, model dilatih menggunakan dataset latih dan dievaluasi menggunakan dataset validasi untuk memantau performa selama pelatihan. Parameter yang digunakan antara lain

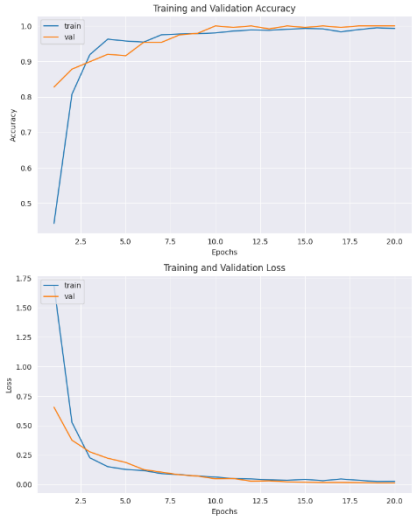
epoch sebanyak 20 dan batch size sebesar 64, yang berarti model memproses seluruh dataset latih sebanyak 20 kali, dengan mempelajari 64 sampel gambar setiap iterasi sebelum memperbarui bobot.

Selain itu, diterapkan dua *callbacks*, yaitu *ModelCheckpoint* dan *EarlyStopping*. *callback ModelCheckpoint* digunakan untuk menyimpan bobot model terbaik berdasarkan akurasi validasi, sedangkan *EarlyStopping* memungkinkan proses pelatihan dihentikan lebih awal jika akurasi tidak mengalami peningkatan signifikan. Hal ini bertujuan untuk menghemat waktu pelatihan dan mengurangi risiko *overfitting*.

Waktu pelatihan juga dicatat menggunakan fungsi time.time() untuk mengetahui total durasi yang dibutuhkan selama proses pelatihan berlangsung. Informasi ini berguna untuk mengevaluasi efisiensi proses pelatihan, khususnya saat membandingkan performa antar model atau konfigurasi pelatihan yang berbeda.

Setelah proses pelatihan selesai, hasilnya dianalisis dengan memvisualisasikan nilai akurasi dan loss pada dataset latih dan validasi dalam bentuk grafik. Grafik ini menunjukkan perubahan akurasi dan loss pada setiap epoch, sehingga dapat digunakan untuk mengevaluasi performa model selama proses pelatihan. Grafik akurasi

menggambarkan kemampuan model dalam mengklasifikasikan data dengan benar, sedangkan grafik loss menunjukkan tingkat kesalahan model dalam melakukan prediksi. Visualisasi hasil pelatihan tersebut ditunjukkan pada Gambar 4.16 berikut ini.



Gambar 4. 16 Hasil Pelatihan Model

Gambar 4.16 menampilkan grafik perubahan akurasi dan loss selama proses pelatihan model. Grafik akurasi menunjukkan peningkatan yang konsisten pada data latih dan validasi hingga mencapai sekitar 100% pada epoch ke-20, sedangkan grafik loss memperlihatkan penurunan nilai loss hingga mendekati nol. Hasil ini menunjukkan bahwa model mampu mempelajari pola data dengan baik dan tidak mengalami *overfitting* maupun *underfitting*.

E. Hasil Evaluasi Model

Pada tahap ini, model yang telah dilatih dievaluasi untuk menguji kemampuannya dalam mengklasifikasikan motif batik dengan baik. Evaluasi dilakukan menggunakan *confusion matrix* dan metrik-metrik evaluasi seperti *accuracy*, *precision*, *recall*, dan *F1-score* untuk memberikan gambaran lebih menyeluruh terhadap performa model. Gambar 4.17 berikut merupakan kode yang digunakan untuk melakukan evaluasi tersebut.

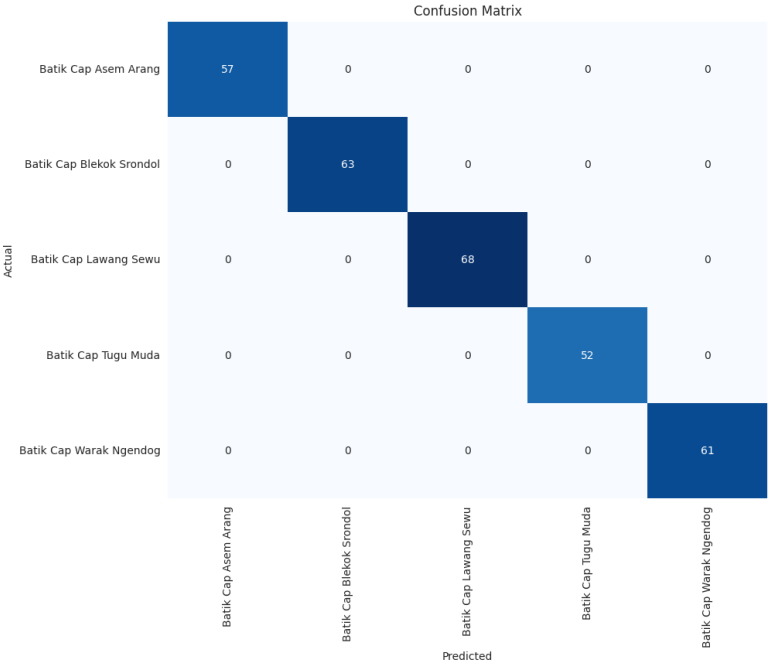
```
from sklearn.metrics import confusion_matrix, classification_report

# Mendapatkan label kelas
y_true = test_data.classes
classes = list(test_data.class_indices.keys())

# Melakukan prediksi pada data uji
preds = model.predict(test_data)
y_pred = np.argmax(preds, axis=1)

# Membuat confusion matrix
cm = confusion_matrix(y_true, y_pred)

# Menampilkan confusion matrix
plt.figure(figsize=(10, 8))
sns.heatmap(cm, annot=True, fmt='g', cmap='Blues', cbar=False,
            xticklabels=classes, yticklabels=classes)
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()

# Menampilkan classification report
report = classification_report(y_true, y_pred, target_names=classes)
print("Classification Report:\n", report)
```

Gambar 4. 17 Kode Evaluasi Model

Kode pada Gambar 4.17 merupakan kode yang digunakan untuk menghasilkan dan menampilkan *confusion matrix* serta *classification report,* yang berisi metrik evaluasi berdasarkan hasil prediksi model pada dataset uji. Pada bagian awal kode, *y_true* berisi kelas asli dari data uji yang diambil dari *test_data.classes*, sementara *classes* menyusun daftar nama kelas yang ada pada dataset menggunakan *test_data.class_indices.keys()*. Selanjutnya, model melakukan prediksi terhadap data uji dengan *model.predict(test_data)*, dan hasil prediksi tersebut disimpan dalam variabel *preds*. Kemudian, untuk mendapatkan kelas yang diprediksi, dilakukan perhitungan menggunakan *np.argmax(preds, axis=1)*, yang memilih indeks kelas dengan probabilitas tertinggi sebagai prediksi model. Setelah itu, *confusion matrix* dihitung dengan menggunakan fungsi *confusion_matrix(y_true, y_pred)* dari *sklearn.metrics*, yang membandingkan kelas asli (*y_true*) dengan kelas prediksi (*y_pred*).

Confusion matrix ini kemudian divisualisasikan dalam bentuk heatmap menggunakan *library seaborn* untuk memudahkan interpretasi hasil klasifikasi model. Setiap baris pada matriks menunjukkan jumlah data sebenarnya dari suatu kelas, sedangkan setiap kolom menunjukkan jumlah prediksi yang diberikan model untuk

kelas tersebut. Nilai pada diagonal utama matriks menunjukkan jumlah prediksi yang benar untuk setiap kelas, sementara nilai di luar diagonal merupakan prediksi yang salah. Dengan demikian, matriks ini memberikan gambaran yang jelas mengenai kinerja model dalam mengklasifikasikan masing-masing kelas. Hasil evaluasi berupa *confusion matrix* tersebut ditampilkan pada Gambar 4.18 berikut ini.



Gambar 4. 18 Hasil *Confusion Matrix*

Berdasarkan hasil *confusion matrix* pada Gambar 4.18, model menunjukkan performa yang sangat baik, di mana seluruh gambar data uji berhasil diklasifikasikan ke dalam kelas yang benar tanpa kesalahan prediksi.

Selain itu, evaluasi performa model juga dilakukan menggunakan fungsi *classification_report()* dari pustaka *sklearn.metrics*, yang menghasilkan metrik seperti *accuracy*, *precision*, *recall*, dan *F1-score*. Hasil *classification report* tersebut ditampilkan pada Gambar 4.19 berikut.

Classification Report:

	precision	recall	f1-score	support
Batik Cap Asem Arang	1.00	1.00	1.00	57
Batik Cap Blekok Srondol	1.00	1.00	1.00	63
Batik Cap Lawang Sewu	1.00	1.00	1.00	68
Batik Cap Tugu Muda	1.00	1.00	1.00	52
Batik Cap Warak Ngendog	1.00	1.00	1.00	61
accuracy			1.00	301
macro avg	1.00	1.00	1.00	301
weighted avg	1.00	1.00	1.00	301

Gambar 4. 19 Hasil *Classification Report*

Berdasarkan Gambar 4.19, nilai *accuracy*, *precision*, *recall*, dan *F1-score* yang dihasilkan adalah 1.00 untuk seluruh kelas batik. Hasil ini menunjukkan bahwa model mampu mengklasifikasikan seluruh data uji secara sempurna tanpa kesalahan prediksi, sehingga dapat disimpulkan bahwa performa model sangat baik dalam mengklasifikasikan motif batik khas Semarang.

F. Hasil Konversi Model

Pada tahap ini, model yang telah dilatih dikonversi ke format *TensorFlow Lite* (TFLite) agar dapat digunakan pada perangkat Android. Konversi ini dilakukan menggunakan *TensorFlow Lite Converter*, yang memungkinkan model berjalan secara optimal di lingkungan mobile tanpa memerlukan koneksi internet.

Proses konversi dilakukan dengan memuat model yang telah dilatih dan menggunakan metode *TFLiteConverter.from_keras_model()*. Model hasil konversi kemudian disimpan dalam file *model_batik.tflite*. Berikut adalah kode untuk melakukan proses konversi:

```
# Load model yang telah dilatih
model = tf.keras.models.load_model("model_batik.h5")

# Konversi model ke format TFLite
converter = tf.lite.TFLiteConverter.from_keras_model(model)
tflite_model = converter.convert()

# Simpan model hasil konversi
with open("model_batik.tflite", "wb") as f:
    f.write(tflite_model)

print("Model berhasil dikonversi ke TensorFlow Lite!")
```

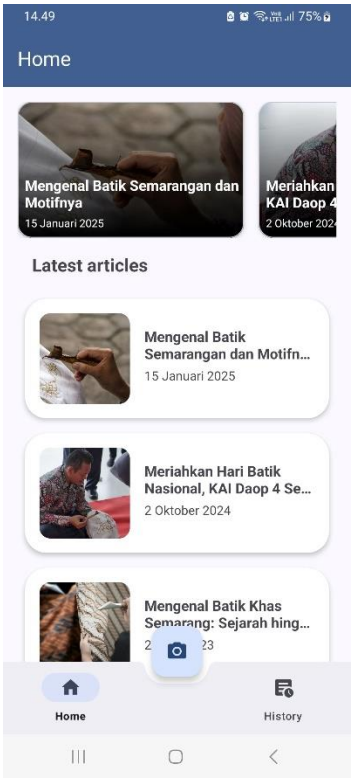
Gambar 4. 20 Kode Konversi Model

Setelah model dikonversi, model TFLite ini kemudian dapat digunakan dalam aplikasi Android untuk melakukan klasifikasi batik secara langsung.

G. Aplikasi

1. Halaman Beranda

Halaman ini merupakan tampilan awal yang muncul saat aplikasi pertama kali dibuka. Pada halaman ini, ditampilkan artikel-artikel terkini mengenai Batik Semarang untuk memberikan informasi terbaru kepada pengguna.



Gambar 4. 21 Halaman Beranda

2. Halaman Klasifikasi

Halaman ini merupakan halaman utama yang digunakan untuk melakukan klasifikasi batik. Pada halaman ini, pengguna dapat memilih untuk mengambil gambar melalui kamera atau memilih gambar dari galeri. Berikut adalah gambar tampilan halaman Klasifikasi.

Gambar 4. 22 Halaman Klasifikasi

3. Halaman Preview

Halaman ini menampilkan gambar batik yang telah dipilih atau diambil oleh pengguna, disertai tombol untuk memulai proses klasifikasi. Pengguna dapat memastikan gambar sudah sesuai sebelum dianalisis lebih lanjut. Berikut adalah gambar tampilan halaman Preview.

Gambar 4. 23 Halaman Preview

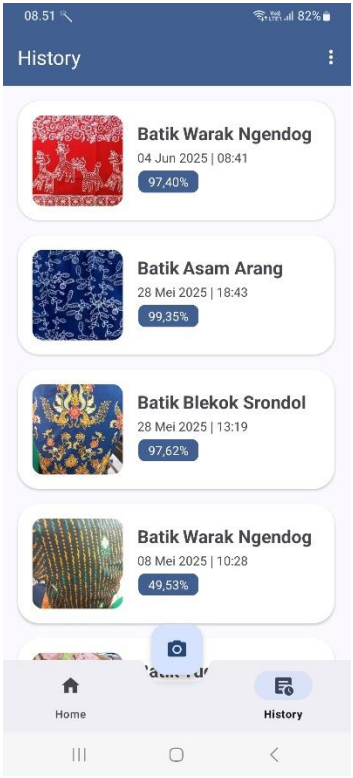
4. Halaman Hasil Klasifikasi

Halaman ini menampilkan hasil klasifikasi batik berdasarkan gambar yang telah dimasukkan oleh pengguna. Informasi yang ditampilkan meliputi gambar, nama motif batik yang dikenali, serta nilai kepercayaan dari model klasifikasi. Berikut adalah gambar tampilan halaman Hasil Klasifikasi.

Gambar 4. 24 Halaman Hasil Klasifikasi

5. Halaman Riwayat

Halaman ini menampilkan daftar hasil klasifikasi batik yang telah dilakukan oleh pengguna. Setiap riwayat mencakup gambar, nama motif batik, nilai kepercayaan, dan waktu klasifikasi. Data riwayat ini disimpan secara lokal dalam aplikasi dan dapat diakses kembali oleh pengguna. Berikut adalah gambar tampilan halaman Riwayat.



Gambar 4. 25 Halaman Riwayat

BAB V
KESIMPULAN DAN SARAN

A. Kesimpulan

Penelitian ini bertujuan untuk membangun model klasifikasi motif batik Semarang menggunakan MobileNetV2 dengan *transfer learning*, mengevaluasi kinerjanya, serta mengimplementasikannya dalam sebuah aplikasi Android.

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan beberapa hal sebagai berikut:

1. Model klasifikasi batik Semarang berhasil dibangun menggunakan arsitektur MobileNetV2 dan pendekatan *transfer learning*. Model ini dilatih menggunakan dataset batik Semarang dan dikonversi ke dalam format *TensorFlow Lite*.
2. Hasil evaluasi model menunjukkan performa yang sangat baik dengan akurasi mencapai 100% pada data uji. Hasil ini menunjukkan bahwa model mampu mengklasifikasikan batik Semarang dengan tepat.
3. Aplikasi Android untuk mengimplementasikan model klasifikasi motif Batik Semarang telah berhasil dibangun dan dapat digunakan untuk melakukan klasifikasi gambar batik secara langsung melalui perangkat.

B. Saran

Berdasarkan hasil penelitian yang telah dilakukan, berikut beberapa saran yang dapat dijadikan bahan pertimbangan untuk pengembangan lebih lanjut:

1. Menambah jumlah motif batik Semarang yang digunakan dalam penelitian, karena pada penelitian ini hanya menggunakan lima jenis motif sehingga cakupan klasifikasinya masih terbatas.
2. Menggunakan arsitektur model lain seperti *Xception*, *EfficientNet*, atau *ResNet* untuk membandingkan kinerja dan akurasi klasifikasi, guna mendapatkan model yang lebih optimal.
3. Menerapkan pendekatan selain *on-device*, seperti menggunakan model yang di-host melalui *cloud*, sehingga pembaruan model dapat dilakukan tanpa perlu memperbarui aplikasi secara keseluruhan serta dapat mengurangi ukuran aplikasi.
4. Meningkatkan tampilan antarmuka aplikasi (UI/UX) agar lebih menarik dan intuitif, sehingga dapat memberikan pengalaman pengguna yang lebih baik.

DAFTAR PUSTAKA

Akhiar, M. (2021). *Penerapan Deep Learning Menggunakan Convolutional Neural Network Pada Klasifikasi Motif Batik*. Universitas Islam Negeri Sultan Syarif Kasim Riau.

Aras, S., Setyanto, A., & Rismayani. (2022). Classification of Papuan Batik Motifs Using Deep Learning and Data Augmentation. *2022 4th International Conference on Cybernetics and Intelligent System (ICORIS)*, 1–5. https://doi.org/10.1109/ICORIS56080.2022.10031320

Imanuel, F., & Tobing, F. A. T. (2023). Implementation of Simple Additive Weighting on Decision Support System for Accoustic Guitar with Web-Based (Study Case: Chaniago Sport). *Ultima InfoSys : Jurnal Ilmu Sistem Informasi*, 100–109. https://doi.org/10.31937/si.v13i2.2963

Intan, N. T. H. A. I., Sugiarto, E., & Wibawanto, W. (2023). Batik Semarang on Cultural Ecology Perspective: Characteristic of Visual Expression. *Catharsis: Journal of Arts Education*, *12*(2), 117–127. https://doi.org/10.15294/catharsis.v12i2.76364

Markoulidakis, I., Rallis, I., Georgoulas, I., Kopsiaftis, G., Doulamis, A., & Doulamis, N. (2021). Multiclass Confusion Matrix Reduction Method and Its Application on Net Promoter Score Classification Problem. *Technologies*, *9*(4), 81. https://doi.org/10.3390/technologies9040081

Negara, B. S., Satria, E., Sanjaya, S., & Dwi Santoso, D. R. (2021). ResNet-50 for Classifying Indonesian Batik with Data Augmentation. *2021 International Congress of Advanced Technology and Engineering (ICOTEN)*, 1–4. https://doi.org/10.1109/ICOTEN52080.2021.9493488

Putra, F. A., Jamil, D. A. C., Prabandanu, B. A., Faruq, S., Pradana, F. A., Alya, R. F., Santoso, H. A., Al Zami, F., & Saputra, F. O. (2021). Classification of Batik Authenticity Using Convolutional Neural Network Algorithm with Transfer Learning Method. *2021 Sixth International Conference on*

Informatics and Computing (ICIC), 1–6. https://doi.org/10.1109/ICIC54025.2021.9632937

Rochmawati, N., Hidayati, H. B., Yamasari, Y., Tjahyaningtijas, H. P. A., Yustanti, W., & Prihanto, A. (2021). Analisa Learning Rate dan Batch Size pada Klasifikasi Covid Menggunakan Deep Learning dengan Optimizer Adam. *Journal of Information Engineering and Educational Technology*, *5*(2), 44–48. https://doi.org/10.26740/jieet.v5n2.p44-48

Samuel, A. L. (1959). Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development*, *3*(3), 210–229. https://doi.org/10.1147/rd.33.0210

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). MobileNetV2: Inverted Residuals and Linear Bottlenecks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4510–4520. https://doi.org/10.1109/CVPR.2018.00474

Sastypratiwi, H., Muhardi, H., & Yulianti, Y. (2024). Batik Recognition and Classification Using Transfer Learning and MobileNet Approach. *JOIV : International Journal on Informatics Visualization*, *8*(4), 2400. https://doi.org/10.62527/joiv.8.4.2407

Uswatun Khasanah, C., Utami, E., & Raharjo, S. (2020). Implementation of Data Augmentation Using Convolutional Neural Network for Batik Classification. *2020 8th International Conference on Cyber and IT Service Management (CITSM)*, 1–5. https://doi.org/10.1109/CITSM50537.2020.9268890

Winarno, E., Septiarini, A., Hadikurniawati, W., & Hamdani, H. (2023). The Hybrid Features and Supervised Learning for Batik Pattern Classification. *Journal on Computing and Cultural Heritage*. https://doi.org/10.1145/3631131

Wiratama, R. W. (2023). *Implementasi dan Klasifikasi Jenis-jenis Batik Menggunakan Algoritma Convolutional Neural*

Network (CNN) Dengan Model Arsitektur ResNet. Politeknik Negeri Malang.

Wona, M. M. A., Aulia Asyifa, S., Virgianti, R., Hamid, M. N., Handoko, I. M., Septiani, N. W. P., & Lestari, M. (2023). Klasifikasi Batik Indonesia Menggunakan Convolutional Neural Network (CNN). *JURTI*, *7*(2), 172–179. https://www.kaggle.com/datasets/dionisiusdh/indonesianbatik-motifs.

LAMPIRAN

Lampiran 1: Lembar Persetujuan Pembimbing

PERSETUJUAN PEMBIMBING

Proposal Skripsi ini telah disetujui oleh Pembimbing untuk dilaksanakan. Disetujui pada:

Hari :

Tanggal :

Pembimbing I	**Pembimbing II**
Hery Mustofa, M.Kom	Dr. Khotibul Umam, S.T., M.Kom
NIP. 198703172019031007	NIP. 197908272011011007

Mengetahui,

Ketua Jurusan Teknologi Informasi

Dr. Khotibul Umam, S.T., M.Kom

NIP. 197908272011011007

Lampiran 2: Lembar Pengesahan Ujian Proposal

KEMENTERIAN AGAMA
UNIVERSITAS ISLAM NEGERI WALISONGO
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Prof Dr. Hamka Kampus III Ngaliyan
Semarang Telp. 7601295 Fax.7615387

PENGESAHAN UJIAN KOMPREHENSIF

Naskah proposal skripsi berikut ini:

Judul : Klasifikasi Motif Batik Khas Semarang Menggunakan *Convolutional Neural Network*
Penulis : Rizky Nur Arifin
NIM : 2108096094
Jurusan : Teknologi Informasi

Telah diujikan dalam sidang komprehensif oleh Dewan Penguji Fakultas Sains dan Teknologi UIN Walisongo Semarang pada Jum'at, 22 November 2024.

Semarang, 23 April 2025

DEWAN PENGUJI

Penguji I,

Dr. Masy Ari Ulinuha, M.T
NIP. 198108122011011007

Penguji II,

Hery Mustofa, M.Kom
NIP. 198703172019031007

Penguji III,

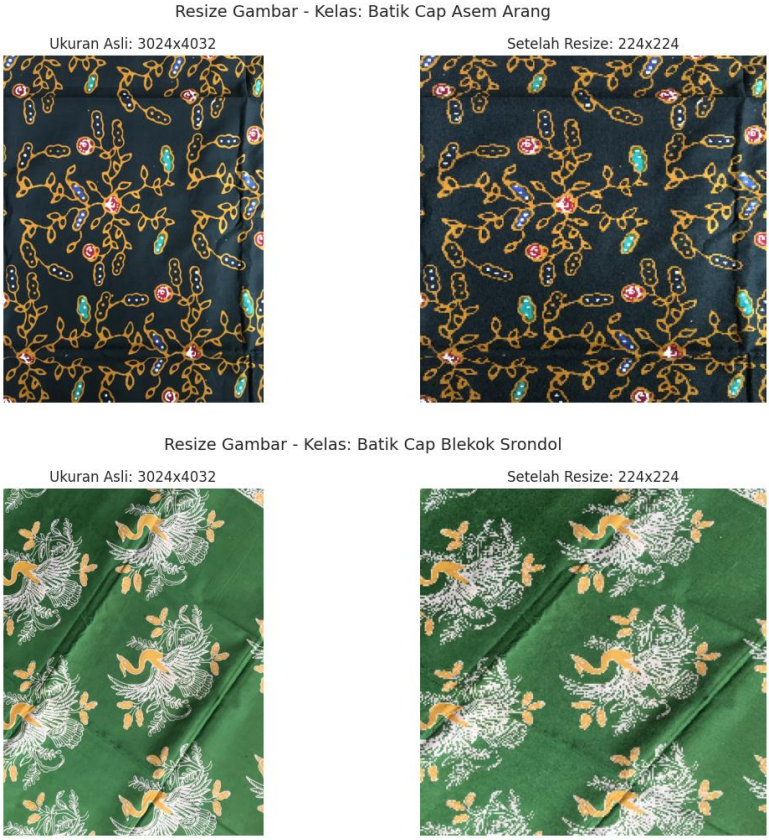
Dr. Wenty Dwi Yuniarti, M. Kom
NIP. 197706222006042002

Penguji IV,

Siti Nur'aini, M.Kom
NIP. 19840131201812001

Lampiran 3: Dataset Batik Semarang

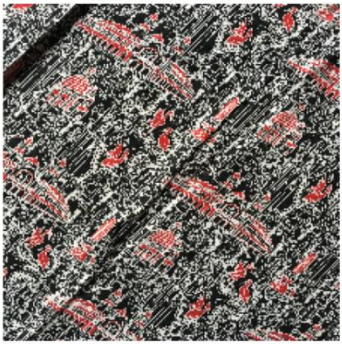
Lampiran 4: Hasil *Resize* Gambar



Resize Gambar - Kelas: Batik Cap Lawang Sewu

Ukuran Asli: 3024x4032

Setelah Resize: 224x224



Resize Gambar - Kelas: Batik Cap Tugu Muda

Ukuran Asli: 3024x4032

Setelah Resize: 224x224

Resize Gambar - Kelas: Batik Cap Warak Ngendog

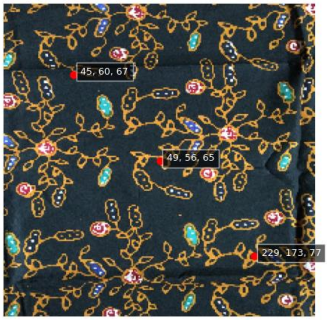
Ukuran Asli: 3024x4032

Setelah Resize: 224x224

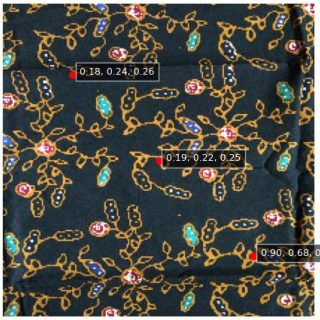
Lampiran 5: Hasil Normalisasi Gambar

Normalisasi Nilai Pixel Gambar - Kelas: Batik Cap Asem Arang

Gambar Sebelum Normalisasi

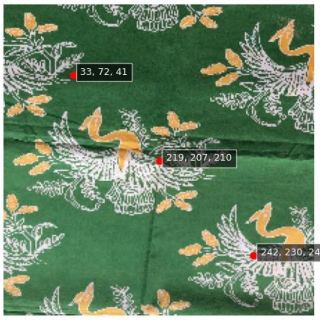


Gambar Setelah Normalisasi (rescale=1./255)

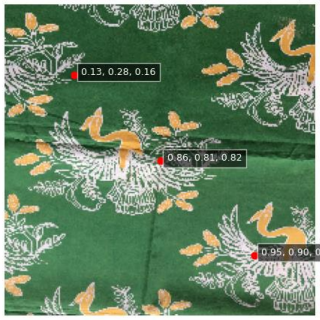


Normalisasi Nilai Pixel Gambar - Kelas: Batik Cap Blekok Srondol

Gambar Sebelum Normalisasi

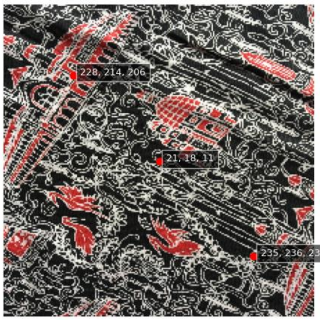


Gambar Setelah Normalisasi (rescale=1./255)



Normalisasi Nilai Pixel Gambar - Kelas: Batik Cap Lawang Sewu

Gambar Sebelum Normalisasi



Gambar Setelah Normalisasi (rescale=1./255)

Normalisasi Nilai Pixel Gambar - Kelas: Batik Cap Tugu Muda

Gambar Sebelum Normalisasi

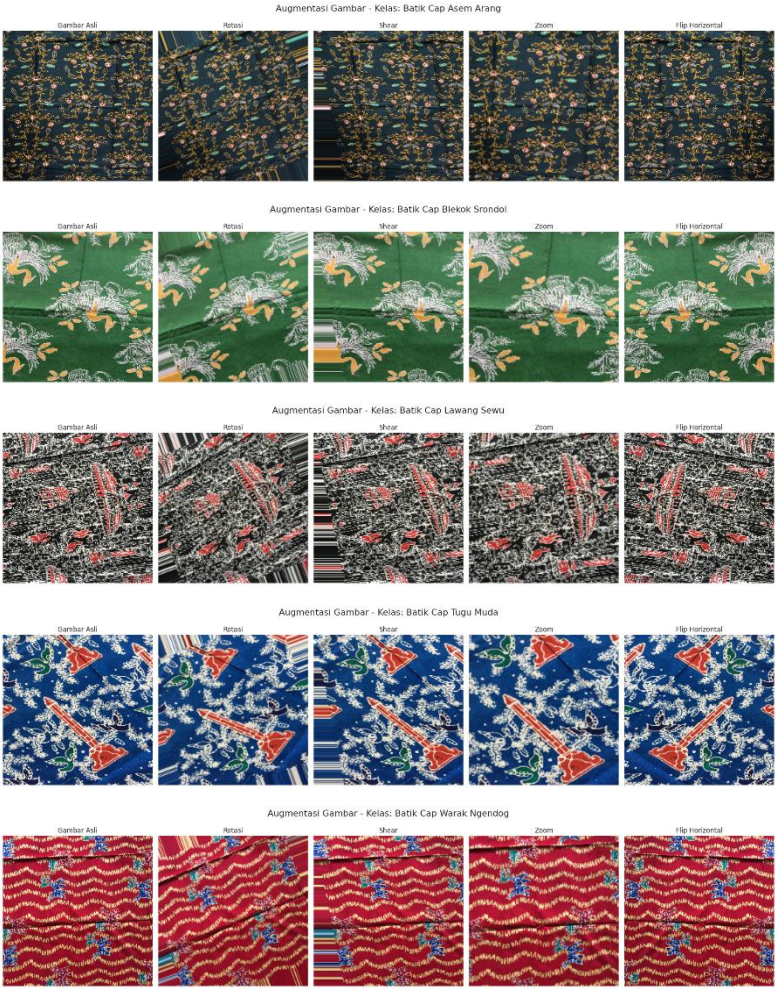
Gambar Setelah Normalisasi (rescale=1./255)

Normalisasi Nilai Pixel Gambar - Kelas: Batik Cap Warak Ngendog

Gambar Sebelum Normalisasi

Gambar Setelah Normalisasi (rescale=1./255)

Lampiran 6: Hasil Augmentasi Gambar



Lampiran 7: Daftar Riwayat Hidup

RIWAYAT HIDUP

A. Identitas Diri

1. Nama Lengkap : Rizky Nur Arifin
2. Tempat & Tanggal Lahir : Bekasi, 02 Juli 2003
3. Alamat Rumah : Perum Grand Cikarang City Blok D-9 No. 33
4. HP : 085972927987
5. Email : arifinnurrizky@gmail.com

B. Riwayat Pendidikan

1. Sekolah Dasar Negeri (SDN) Karang Baru 02
2. Sekolah Menengah Pertama Islam Terpadu (SMPIT) El-Hurriyah
3. Madrasah Aliyah Negeri (MAN) 1 Bekasi