**ANOTASI KARAKTER PADA VIDEO WAYANG KULIT DENGAN METODE *CANNY EDGE DETECTION* DAN ALGORITMA *CONVOLUTIONAL NEURAL NETWORK (CNN)***

**TUGAS AKHIR**



Oleh:

SANDING RIYANTO

**NIM : 1811501541**

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS BUDI LUHUR

JAKARTA

**2022**

**ANOTASI KARAKTER PADA VIDEO WAYANG KULIT DENGAN METODE *CANNY EDGE DETECTION* DAN ALGORITMA *CONVOLUTIONAL NEURAL NETWORK (CNN)***

**Diajukan untuk memenuhi salah satu persyaratan memperoleh gelar Sarjana Komputer (S. Kom)**

**TUGAS AKHIR**



Oleh:

SANDING RIYANTO

**NIM : 1811501541**

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS BUDI LUHUR

JAKARTA

**2022**

# LEMBAR PENGESAHAN

# ABSTRAK

# KATA PENGANTAR

Alhamdulillah, segala puji dan syukur penulis panjatkan atas ke hadirat Allah SWT, karena berkat nikmat dan karunianya, maka penulis bisa menyelesaikan Tugas Akhir ini dengan baik. Adapun Tugas Akhir ini disusun untuk memenuhi persyaratan dalam pendidikan Strata Satu (S1) pada Program Studi Teknik Informatika, Fakultas Teknologi Informasi Universitas Budi Luhur.

Dalam kesempatan kali ini, tak lupa penulis ingin mengucapkan terima kasih kepada pihak yang telah memberikan bimbingan dan masukan berupa kritik dan saran dalam penyusunan Tugas Akhir ini. Oleh karena itu, penulis menyampaikan rasa hormat dan ucapan terima kasih kepada:

1. Allah SWT, atas segala Petunjuk dan Kemudahan-Nya sehingga pada akhirnya penulis dapat menyelesaikan Laporan Tugas Akhir.
2. Orang tua, dan saudara dari penulis yang telah memberikan dukungan lahir maupun batin kepada penulis selama pembuatan Tugas Akhir.
3. Bapak Dr. Ir. Wendi Usino, M.Sc., MM, selaku Rektor Universitas Budi Luhur.
4. Bapak Dr. Deni Mahdiana, S.Kom., M.M, M.Kom, selaku Dekan Fakultas Teknologi   
   Informasi Universitas Budi Luhur.
5. Bapak Dr. Indra, S.Kom, M.T.I, selaku Ketua Program Studi Teknik   
   Informatika, Fakultas Teknologi Informasi Universitas Budi Luhur.
6. Bapak Rizky Pradana, S.Kom., M.Kom, selaku Sekretaris Program Studi Teknik Informatika dan sekaligus Dosen Penasihat Akademik, Fakultas Teknologi Informasi, Universitas Budi Luhur.
7. Bapak Dr. Achmad Solichin, S.Kom., M.T.I, selaku Dosen Pembimbing Tugas Akhir yang telah memberi arahan dan juga dukungandalam proses pembuatan aplikasi.
8. Rekan-Rekan Asisten Lab. ICT Terpadu yang setia dan kompak dalam segala hal, dari awal sampai akhir proses pembuatan Tugas Akhir ini.
9. Rekan-Rekan KUTI-2018 yang selama ini membantu dan saling mendukung dalam kegiatan perkuliahan.
10. Semua pihak yang telah terlibat namun tidak dapat penulis sebutkan satu persatu. Semoga Allah SWT melimpahkan rahmat dan karunia-Nya atas segala bantuan yang telah diberikan. Aamiin.

Akhir kata penulis memohon maaf atas segala kekurangan dan kekeliruan yang ada pada laporan Tugas Akhir ini. Semoga tulisan ini dapat bermanfaat bagi banyak pihak.

Jakarta, ….. 2022

Penulis

# DAFTAR TABEL

# DAFTAR GAMBAR

# DAFTAR ISI

Halaman

[LEMBAR PENGESAHAN 3](#_Toc106484599)

[ABSTRAK 4](#_Toc106484600)

[KATA PENGANTAR 5](#_Toc106484601)

[DAFTAR TABEL 6](#_Toc106484602)

[DAFTAR GAMBAR 7](#_Toc106484603)

[DAFTAR ISI 8](#_Toc106484604)

[BAB I PENDAHULUAN 11](#_Toc106484605)

[1.1. Latar Belakang 11](#_Toc106484606)

[1.2. Masalah 12](#_Toc106484607)

[1.3. Batasan Masalah 12](#_Toc106484608)

[1.4. Tujuan Penulisan 12](#_Toc106484609)

[1.5. Manfaat 12](#_Toc106484610)

[1.5.1. Manfaat bagi penulis: 12](#_Toc106484611)

[1.5.2. Manfaat bagi institusi Universitas Budi Luhur: 12](#_Toc106484612)

[1.5.3. Manfaat bagi dunia ilmu pengetahuan: 13](#_Toc106484613)

[1.6. Sistematika Penulisan 13](#_Toc106484614)

[BAB II LANDASAN TEORI 14](#_Toc106484615)

[2.1. Anotasi 14](#_Toc106484616)

[2.2. Wayang Kulit 14](#_Toc106484617)

[2.3. Motion Detection 14](#_Toc106484618)

[2.4. OpenCV 15](#_Toc106484619)

[2.5. Edge Detection 15](#_Toc106484620)

[2.6. Artificial Intellegence 15](#_Toc106484621)

[2.7. Machine Learning 16](#_Toc106484622)

[2.7.1. Algoritma *Machine Learning* 16](#_Toc106484623)

[*2.7.2. Deep Learning* 17](#_Toc106484624)

[2.8. Convolutional Neural Network (CNN) 17](#_Toc106484625)

[*2.8.1. Convolution* 18](#_Toc106484626)

[*2.8.2. Subsampling* 18](#_Toc106484627)

[*2.8.3. Activation* 18](#_Toc106484628)

[*2.8.4. Fully Connected* 18](#_Toc106484629)

[2.9. MobileNet-v2 19](#_Toc106484630)

[2.10. Studi Literatur 19](#_Toc106484631)

[BAB III METODOLOGI PENELITIAN 26](#_Toc106484632)

[3.1. Data Penelitian 26](#_Toc106484633)

[3.2. Penerapan Metode 27](#_Toc106484634)

[3.2.1. Pengumpulan Data 28](#_Toc106484635)

[*3.2.2. Preprocessing* 29](#_Toc106484636)

[3.2.3. Pembagian Data 30](#_Toc106484637)

[3.2.4. *Modeling* 31](#_Toc106484638)

[3.2.5. Identifikasi Data Uji 37](#_Toc106484639)

[3.2.6. Visualisasi Hasil Identifikasi 37](#_Toc106484640)

[3.3. Rancangan Layar Aplikasi 37](#_Toc106484641)

[3.3.1. Rancangan Layar Menu Utama 38](#_Toc106484642)

[3.3.2. Rancangan Layar *About Application* 38](#_Toc106484643)

[3.3.3. Rancangan Layar *Preprocessing Images* 38](#_Toc106484644)

[3.3.4. Rancangan Layar *Training CNN Model* 39](#_Toc106484645)

[3.3.5. Rancangan Layar *Show Image-Video* 39](#_Toc106484646)

[3.3.6. Rancangan Layar *Identify Image* 40](#_Toc106484647)

[3.3.7. Rancangan Layar *Identify Video* 40](#_Toc106484648)

[3.3.8. Rancangan Layar *Split Dataset* 41](#_Toc106484649)

[3.3.9. Rancangan Layar *Message Box* 41](#_Toc106484650)

[3.4. Rancangan Pengujian 42](#_Toc106484651)

[3.4.1. Akurasi 42](#_Toc106484652)

[3.4.2. Presisi 42](#_Toc106484653)

[3.4.3. *Recall* 42](#_Toc106484654)

[BAB IV HASIL DAN PEMBAHASAN 44](#_Toc106484655)

[4.1. Lingkungan Percobaan 44](#_Toc106484656)

[4.1.1. Spesifikasi perangkat keras 44](#_Toc106484657)

[4.1.2. Spesifikasi perangkat lunak 44](#_Toc106484658)

[4.2. Implementasi Metode 44](#_Toc106484659)

[4.2.1. Tahap Pengumpulan Data 44](#_Toc106484660)

[4.2.2. Tahap *Preprocessing* 44](#_Toc106484661)

[4.2.3. Tahap Pembagian Data 55](#_Toc106484662)

[4.2.4. Tahap *Modeling* 55](#_Toc106484663)

[4.2.5. Tahap Identifikasi Data Uji 55](#_Toc106484664)

[4.2.6. Visualisasi Hasil Identifikasi 55](#_Toc106484665)

[4.3. Flowchart Tahapan Metode 55](#_Toc106484666)

[4.3.1. Flowchart Keseluruhan Sistem 55](#_Toc106484667)

[4.3.2. Flowchart Preprocessing Image 55](#_Toc106484668)

[4.3.3. Flowchart Training CNN Model 55](#_Toc106484669)

[4.3.4. Flowchart Images Augmentation 55](#_Toc106484670)

[4.3.5. Flowchart Identifikasi Image 55](#_Toc106484671)

[4.3.6. Flowchart Identifikasi Video 55](#_Toc106484672)

[4.4. Algoritma Tahapan Metode 55](#_Toc106484673)

[4.4.1. Algo1 55](#_Toc106484674)

[4.4.2. Algo 2 55](#_Toc106484675)

[4.5. Pengujian 55](#_Toc106484676)

[4.6. Tampilan Layar Aplikasi 55](#_Toc106484677)

[4.6.1. Tampilan Layar Menu Utama 55](#_Toc106484678)

[4.6.2. Dst 55](#_Toc106484679)

[BAB V PENUTUP 56](#_Toc106484680)

[5.1. Kesimpulan 56](#_Toc106484681)

[5.2. Saran 56](#_Toc106484682)

[LAMPIRAN 57](#_Toc106484683)

# PENDAHULUAN

## Latar Belakang

Wayang Kulit merupakan sebuah bentuk pertunjukan teater boneka dari Jawa Tengah dan Jawa Timur (suku *Jawa*), yang menggunakan figur-figur tokoh yang dibuat dari kulit kerbau, atau terkadang kulit sapi. Pertunjukan wayang dapat dipertontonkan bayangannya dari balik kelir untuk memperlihatkan detail *tatahan*-nya, tetapi juga dapat dipertontonkan dari depan kelir untuk memperlihatkan keindahan seni *sunggingan*-nya. Cerita atau *lakon* yang disuguhkan berhulu dari Kitab *Ramayana* dan *Mahabharata*. Dalam setiap ­*lakon* lazimnya memperlihatkan konflik antara tokoh wayang yang baik (protagonis) dan yang berwatak jahat (antagonis).

Wayang kulit menjadi salah satu kebudayaan asli Indonesia yang telah diakui oleh UNESCO sebagai *Masterpiece of Oral and Intangible Heritage of Humanity* atau karya kebudayaan yang mengagumkan di bidang cerita narasi dan warisan budaya yang indah dan berharga sejak 7 November 2003. Hal ini menjadi motivasi sekaligus perhatian bagi seluruh elemen bangsa agar terus melestarikan dan menjaga substansi dari seni wayang kulit ini. Salah satu caranya yaitu dengan berusaha mengenal dan memahami karakter yang ada pada wayang kulit.

Namun, sebagian besar generasi muda saat ini tidak mengenal macam-macam karakter tokoh wayang kulit, hanya generasi yang lahir sebelum tahun 90-an. Ditambah ragam bentuk pada wayang kulit yang mempunyai kemiripan, sehingga orang awam akan sulit untuk membedakannya. Meskipun sejak adanya pandemi Covid-19, praktek pementasan wayang kulit bisa ditonton melalui video di *YouTube*, tetapi hal ini hanya sebuah solusi terkait mekanisme pagelaran wayang kulit, belum menyelesaikan masalah penonton yang tidak mengenal jenis karakter wayang kulit. Hal ini akan menjadi masalah ketika seluruh lapisan masyarakat sudah tidak ada lagi yang mengenal nama dan karakter tokoh wayang kulit ini.

Berdasarkan uraian di atas, penulis memberikan alternatif solusi untuk membuat sistem aplikasi yang dapat menentukan jenis karakter wayang kulit dari sebuah video. Sistem aplikasi ini mengadopsi salah satu cabang dari teknologi *Artificial Intelligence*, aitu *Computer Vision* khususnya *Video Classification*.

## Rumusan Masalah

Berdasarkan uraian latar belakang di atas, maka dapat disimpulkan rumusan masalah sebagai berikut:

Bagaimana menerapkan metode *Canny Edge Detection* dan algoritma *Convolutional Neural Network (CNN)* untuk identifikasi nama-nama karakter wayang kulit?

Berapa nilai akurasi yang diperoleh dari model algoritma *Convolutional Neural Network (CNN)* dalam melakukan identifikasi nama-nama karakter wayang kulit?

## Batasan Masalah

Adapun Batasan atau ruang lingkup masalah pada penelitian ini adalah sebagai berikut:

*Platform* yang digunakan berupa aplikasi desktop.

Jumlah karakter wayang kulit yang dijadikan sebagai *dataset* hanya karakter-karakter tertentu, antara lain: *abimanyu, anoman, arjuna, bagong, baladewa bima,buta, cakil, durna, dursasana, duryudana, gareng, gatotkaca, karna, kresna, nakula-sadewa, patih sabrang, petruk, puntadewa, semar, sengkuni, dan togog.*

Video uji dibatasi berdurasi kurang dari 1 menit.

## Tujuan Penulisan

Adapun tujuan dari dilakukannya penelitian ini, yaitu:

Menerapkan metode *Canny Edge Detection* pada proses preprocessing dataset citra wayang kulit.

Merancang sebuah *model* dari algoritma *Convolutional Neural Network (CNN)* untuk menentukan nama karakter wayang kulit.

Menguji keakuratan algoritma *Convolutional Neural Network (CNN)* dalam melakukan identifikasi citra wayang kulit.

## Manfaat

Adapun manfaat dari penelitian ini adalah masyarakat awam khususnya para generasi muda akan terbantu dalam mengenali nama-nama karakter wayang kulit yang jumlahnya sangat banyak. Sehingga, esensi dan substansi nilai moral maupun nilai-nilai yang lain pada kesenian wayang kulit bisa diperoleh. Hasil penelitian ini juga diharapkan menjadi penyemangat bagi generasi milenial untuk dapat ikut serta melestarikan kesenian wayang kulit di era teknologi seperti sekarang ini.

## Sistematika Penulisan

Sistematika penulisan penelitian ini disusun untuk memberikan gambaran umum tentang penelitian yang dijalankan. Sistematika penulisan tugas akhir ini adalah sebagai berikut:

**BAB I : PENDAHULUAN**

Bagian ini berisi tentang latar belakang, rumusan masalah, batasan masalah, manfaat dan tujuan penelitian, dan juga membahas mengenai sistematika penulisan.

**BAB II : LANDASAN TEORI**

Bagian ini berisi tentang algoritma dan metode yang akan dibahas, serta teori-teori yang berkaitan dengan penelitian ini, yaitu pengertian dan pemahaman anotasi, wayang kulit, *motion detection, OpenCV, Edge Detection, Artificial Intellegence, Machine Learning*, *Convolutional Neural Network (CNN)*, dan Studi Literatur.

**BAB III : METODOLOGI PENELITIAN**

Bagian ini berisi tentang sumber data penelitian, penerapan atau tahapan metode yang digunakan. Bab ini juga berisi tentang rancangan pengujian dari ekstraksi informasi yang didapat.

**BAB IV : HASIL DAN PEMBAHASAN**

Bagian ini berisi mengenai lingkungan percobaan sistem yang dibuat, implementasi metode, *flowchart* tahapan metode, dan uraian algoritme pada proses, serta analisa pengujian sistem yang telah dibangun apakah data hasil pengelompokan yang didapat sudah sesuai dan relevan

**BAB V : PENUTUP**

Bagian ini berisi tentang kesimpulan yang dapat ditarik dari penelitian dan saran untuk pengembangan lebih lanjut mengenai topik terkait dalam penelitian berikutnya.

# LANDASAN TEORI

## Wayang Kulit

Wayang kulit *purwa* adalah sebuah bentuk pertunjukan teater boneka dari Jawa Tengah dan Jawa Timur (suku *Jawa*), yang menggunakan figur-figur tokoh yang dibuat dari kulit kerbau, atau terkadang kulit sapi. Pertunjukan wayang dapat dipertontonkan bayangannya dari balik kelir untuk menunjukkan detail *tatahan*-nya yang rumit, tetapi juga bisa dipertontonkan dari depan kelir, *sunggingan*-nya yang dikerjakan dengan teliti. Kata *purwa* mengacu kepada ragam lakon yang dipakai dalam kesenian ini, yang meliputi cerita-cerita epos Mahabharata dan Ramayana, juga lakon Jawa-Dewa, Lokapala, dan Arjusasrabahu (Emerson, 2017).

Menurut para ahli, selain sebagai pertunjukan dan hiburan, wayang juga berguna untuk membentuk watak dan karakter seseorang atau penontonnya (Resa Arif Yudianto et al., 2020). Sehingga hal ini menjadi penting untuk dilestarikan dan dijaga substansinya, agar generasi penerus bangsa dapat mengenal dan memahami berbagai macam karakter dari wayang kulit.

Adapun beberapa contoh karakter wayang kulit yang cukup populer di kalangan kita, antara lain: *Abimanyu, Anoman, Arjuna, Bagong, Bima, Baladewa, Buta, Cakil, Durna, Duryudana, Dursasana, Gareng, Gatotkaca, Karna, Kresna, Nakula-Sadewa, Petruk, Puntadewa, Semar, Sengkuni, dan Togog.*

## Motion Detection

Deteksi gerakan atau *Motion Detection*, yang merupakan langkah mendasar dalam pengawasan video, bertujuan untuk mendeteksi wilayah sesuai dengan benda bergerak. Informasi yang dihasilkan sering menjadi dasar untuk tingkat yang lebih tinggi operasi yang membutuhkan hasil yang tersegmentasi dengan baik, seperti klasifikasi objek dan tindakan atau aktivitas pengakuan. Namun, *motion detection* mengalami masalah yang disebabkan oleh kebisingan sumber, kompleks latar belakang, variasi pencahayaan pemandangan, dan bayangan objek statis dan bergerak. Berbagai metode telah diusulkan untuk mengatasi masalah ini dengan hanya mempertahankan gerakan objek yang menarik (Sehairi et al., 2017).

## OpenCV

*OpenCV* *(Open Source Computer Vision Library)* merupakan *open source computer vision* dan perpustakaan perangkat lunak pembelajaran mesin atau *machine learning*. OpenCV dibangun untuk menyediakan infrastruktur umum untuk aplikasi computer vision, dan untuk mempercepat penggunaan persepsi mesin dalam produk komersial. Menjadi produk berlisensi BSD, OpenCV memudahkan bisnis untuk memanfaatkan dan memodifikasi kode.

*OpenCV* memiliki antarmuka C++, Python, Java dan MATLAB dan mendukung sistem operasi Windows, Linux, Android dan Mac OS. *OpenCV* sebagian besar condong ke *computer vision* waktu nyata dan memanfaatkan instruksi MMX dan SSE bila tersedia. Antarmuka CUDA dan OpenCL berfitur lengkap sedang dikembangkan secara aktif saat ini. Ada lebih dari 500 algoritme dan sekitar 10 kali lebih banyak fungsi yang menyusun atau mendukung algoritme tersebut. OpenCV ditulis secara native dalam C++ dan memiliki antarmuka templat yang berfungsi mulus dengan wadah STL.

*OpenCV* menjadi alat yang populer untuk mengembangkan *vision applications*, karena membutuhkan pengetahuan yang signifikan tentang alat dan teknik, yang terdiri dari algoritma yang paling umum dari *computer vision*. Untuk teknik yang lebih baru seperti pembelajaran mendalam, Keras dan TensorFlow dapat digunakan dalam membuat aplikasi (Dadhich, 2018).

## Edge Detection

Deteksi tepi *(edge detection)* pada suatu citra adalah suatu proses yang menghasilkan tepi-tepi dari objek-objek gambar. Suatu titik (x, y) dikatakan sebagai tepi *(edge)* dari suatu citra bila titik tersebut mempunyai perbedaan yang tinggi dengan tetangga. Pendeteksian tepi citra berfungsi untuk memperoleh tepi objek. Deteksi tepi memanfaatkan perubahan nilai intensitas yang drastis pada batas dua area (Sukatmi, 2017).

Terdapat beberapa jenis fungsi deteksi tepi, yaitu: Prweitt, Sobel, dan Canny. Dalam penelitian ini, penulis menggunakan metode Canny Edge Detection. Pada dasarnya, metode deteksi tepi Canny digunakan untuk menghilangkan noise dan operasi filetring serta segmentasi citra. Dalam tahapannya deteksi tepi Canny mempunyai tahapan: (1) Smoothing citra, (2) Menghiung gradien citra, (3) Menghitung supresi maksimal, (4) Operasi *thresholding* (Susanto et al., 2017).

## Artificial Intellegence

*Artificial Intellegence* (AI) atau Kecerdasan Buatan adalah studi tentang teori dan pengembangan sistem komputer agar mampu melakukan tugas-tugas yang dahulu hanya dapat dilakukan oleh manusia. Seperti membedakan berbagai gambar, menjawab pertanyaan, mengenali dan menerjemahkan bahasa, dan lain-lain (Primartha, 2018).

Menurut John McCarthy, salah satu pakar *Artificial Intellegence*, AI adalah *“the science and engineering of making intelligent machines.”*. Berikut adalah definisi AI yang lain:

Sebuah cabang dari ilmu komputer *(computer science)* yang berkaitan dengan bagaimana mensimulasikan kecerdasan pada komputer.

Kemampuan mesin untuk meniru kecerdasan manusia.

Sebuah sistem komputer yang mampu melakukan pekerjaan dimana pekerjaan tersebut memerlukan kecerdasan manusia, seperti *visual perception, speech recognition, decision-making,* dan penerjemahan bahasa.

Dalam perkembangannya, teknologi kecerdasan buatan atau AI ini mulai menuju ke arah yang lebih spesifik, yaitu dengan munculnya konsep *Machine Learning*.

## Machine Learning

*Machine Learning* adalah bidang ilmu pengetahuan teknologi yang memberi komputer kemampuan untuk belajar tanpa diprogram secara eksplisit. *Machine learning* memerlukan sebuah model yang didefinisikan berdasarkan parameter-parameter tertentu. Proses *learning* adalah eksekusi program komputer untuk mengoptimasi parameter-parameter dari model tersebut, dengan memanfaatkan data latih dan *past experience.* Jadi, secara sederhana dapat dijelaskan bahwa *machine learning* merupakan pemrograman computer untuk mencapai kriteria/performa tertentu dengan menggunakan sekumpulan data latih atau pengalaman di masa lalu *(past experience)* (Primartha, 2018).

### Algoritma *Machine Learning*

Dalam matematika dan ilmu komputer, algoritma adalah urutan atau langkah-langkah untuk perhitungan atau menyelesaikan suatu masalah yang ditulis secara berurutan. Sehingga, algoritma pemrograman adalah urutan atau langkah-langkah untuk menyelesaikan masalah pemrograman komputer (Primartha, 2018).

Secara umum algoritma machine learning dapat dikelompokkan sebagai berikut:

*Supervised learning*

*Unsupervised learning*

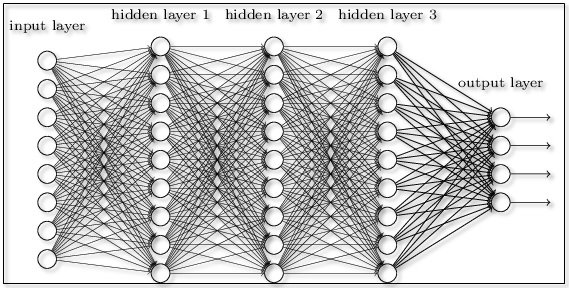
*Reinforcement learning*

*Deep learning*

Pada sumber yang lain, algoritma machine learning dikelompokkan ke dalam tiga jenis, yaitu: *supervised learning, unsupervised learning, dan reinforcement learning* (Raschka & Mirjalili, 2017). Pada kenyataannya, jumlah algoritma machine learning sangat banyak jumlahnya, tetapi rata-rata ahli mengelompokkan ke dalam tiga atau empat algoritma.

### *Deep Learning*

Deep learning merupakan metode learning yang memanfaatkan artificial neural networks yang berlapis-lapis (multi-layer). Artificial neural networks ini dibuat mirip dengan otak manusia, di mana neuron-neuron terkoneksi satu sama lain sehingga membentuk sebuah jaringan neuron yang sangat rumit. *Deep Learning* atau *Deep Structured Learning* atau *Hierarchial Learning* atau *Deep Neural* merupakan metode learning yang memanfaatkan *Multiple Non-Linear Transformation* (Primartha, 2018).



**Gambar 2.1 Ilustrasi *Multilayer Neural Networks***

Pada Gambar 2.1 merupakan diagram ilustrasi dari konsep Multilayer Neural Networks atau MLP. Secara prinsip, MLP adalah pengembangan dari konsep SLP *(Single Layer Perceptron)*, jika SLP relevan digunakan untuk pengelompokkan 2 kelas, maka MLP bisa *multiple class*. MLP menjadi cikal bakal metode *Deep Learning*, yang cara kerjanya meniru otak manusia dalam hal mengirimkan informasi dari satu neuron ke neuron yang lain.

Beberapa algoritma yang termasuk dalam kategori *Deep Learning* yaitu:

*Convolutional Neural Network* (CNN)

*Restricted Boltzmann Machine* (RBM)

*Deep Belief Networks* (DBN)

*Stacked Autoencoders*

## Convolutional Neural Network (CNN)

Algoritma *Convolutional Neural Network* adalah sebuah MLP *(Multi-Layer Perceptron)* yang didesain secara khusus untuk mengidentifikasi image/citra/gambar dua dimensi. CNN meniru cara kerja otak manusia untuk mengenali objek yang dilihatnya (Primartha, 2018). Sedangkan pada pengertian lain, *Convolutional Neural Network* (CNN) adalah algoritma klasifikasi hasil dari pengembangan *Multilayer Perceptron* (MLP) yang dibangun untuk memproses data dua dimensi. Karena kedalaman jaringan yang tinggi dan banyak diterapkan pada data gambar/citra sehingga CNN termasuk dalam dari *Deep Neural Network* (Resa Arif Yudianto et al., 2020).

Secara garis besar, mekanisme kerja algoritma CNN dapat dikelompokkan ke dalam 4 tahap/bagian, yaitu (Primartha, 2021):

### *Convolution*

*Convolution filters* atau *Convolution layers* adalah layer-layer pertama yang berfungsi untuk menerima *input* (*input signal*). Atau dengan kata lain, proses convolution yaitu sebuah proses pada CNN di mana *network* mencoba untuk memberi label pada sebuah sinyal *input* yang berdasarkan dengan apa yang sudah dipelajari pada masa lalu. Secara umum, konvolusi merupakan operasi antardua fungsi, sehingga menghasilkan fungsi ketiga yang merupakan hasil modifikasi dari kedua fungsi sebelumnya.

### *Subsampling*

*Subsampling* merupakan proses memperhalus data input yang sudah melalui tahap *Convolution Layer* dengan tujuan untuk mengurangi tingkat sensifitas terhadap *noise* dan *variations*. Contoh *subsampling* yaitu dengan mengurangi ukuran gambar, atau mengurangi kontras warna (RGB *channel*).

### *Activation*

*Layer activation* berfungsi untuk mengendalikan bagaimana sinyal mengalir dari satu layer ke layer berikutnya, sedemikian rupa mnegikuti cara kerja neuron-neuron. Sehingga, sinyal output yang terkait dengan referensi sebelumnya akan mengaktifkan banyak neuron yang lain, dan sinyal dapat dipropagasi secara efisien.

CNN kompatibel dengan berbagai *activation function* yang kompleks. Activation yang paling umum yaitu *Rectified Linear Unit* (ReLU). Namun, pada permasalahan *multiclass classification*, *output layer* memiliki lebih dari 1 neuron, sehingga fungsi aktivasi yang sering digunakan adalah *softmax*. *Softmax activation* merupakan bentuk lain dari algoritma *Logistic Regression* yang dapat digunakan untuk mengklasifikasi lebih dari dua kelas (Wulandari et al., 2020).

### *Fully Connected*

*Fully Connected Layer* adalah *Multi-Layer Perceptron* (MLP) yang menggunakan *Softmax Activation Function.* Kata “*Fully Connected*” menunjukkan bahwa setiap neuron pada *layer* sebelumnya sudah terkoneksi dengan setiap neuron pada *layer* berikutnya. Tujuan dari *Fully Connected Layer* yaitu untuk klasifikasi (*class*) *image* atau sejenisnya.

## MobileNet-v2

MobileNet-v2 adalah jaringan saraf *convolutional* yang memiliki kedalaman 53 lapisan. Pada jaringan MobileNet-v2 memiliki arsitektur yang ringan sehingga memungkinkan data diproses lebih cepat dibandingkan dengan arsitektur CNN lainnya seperti VGG, ResNet, dan lain sebagainya (Indraswari et al., 2022). MobileNet-v2 masih menggunakan konvolusi yang mendalam dan terarah seperti MobileNet-v1. MobileNet-v2 menambahkan dua fitur baru: 1) *Linear Bottlenecks*, dan 2) *Shortcut Connections Between Bottlenecks* (Farid Naufal & Ferdiana Kusuma, 2021). Untuk menentukan kondisi optimum dari arsitektur MobileNet-v2, ada beberapa pilihan optimizer yang dapat digunakan, diantaranya *Stochastic Gradient Descent* (SGD), RMSProp, dan ADAM (Wikarta et al., 2021).

## Studi Literatur

**Tabel 2.1 Studi Literatur**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **JUDUL** | **TUJUAN PENELITIAN** | **METODE** | **HASIL PENELITIAN** |
| 1 | Rekognisi Wayang Kulit Menggunakan Jaringan Syaraf Tiruan (Susanto et al., 2019) | Mengenali beragam nama-nama wayang kulit berdasarkan pengolahan citra digital. | Deteksi tepi dan Jaringan Syaraf Tiruan. | Dari 100 kali percobaan, terdapat 4 kali percobaan yang belum dapat mengenali gambar wayang dengan benar. Dengan demikian dapat disimpulkan bahwa pada penelitian ini, telah di hasilkan akurasi rekognisi wayang sebesar 96%. |
| 2 | Analisis Pengaruh Tingkat Akurasi Klasifikasi Citra Wayang Dengan Algoritma *Convolutional Neural Network (CNN)* (Resa Arif Yudianto et al., 2020) | Untuk mengetahui pengaruh dari tingkat akurasi klasifikasi (citra wayang kulit) yang dihasilkan menggunakan algoritma CNN. | CNN | Nilai akurasi sebesar 97%, *loss/error* 2%, presisi 93% dan *recall* sebesar 87%. Penggunaan *augmentation* dan pengubahan *channel* citra ke *grayscale* pada *preproccessing* data *training* sangat berpengaruh signifikan terhadap nilai akurasi model yang dihasilkan dikombinasikan dengan rasio *dataset* 80:20 dan *epoch* 100. Penggunaan nilai *epoch* juga berpengaruh terhadap akurasi model, semakin tinggi nilai *epoch* nilai akurasi yang dihasilkan semakin baik jika *dataset* yang digunakan lebih banyak. |
| 3 | Klasifikasi Citra Wayang Dengan Menggunakan Metode k-NN & GLCM (Sandy et al., 2019) | Mendeteksi dan mengklasifikasi objek (wayang) menurut bentuk dari objek (wayang) tersebut, sehingga dapat dikenali oleh sistem berdasarkan model yang telah dibuat. | k-NN dan GLCM | Berdasarkan hasil penelitian yang telah dilakukan, tingkat akurasi sudah cukup bagus, terutama pada pengenalan pola wayang Bathara Wisnu dengan tingkat akurasi 90%, namun rendah saat proses pengenalan pola wayang Yudhistira, yaitu 60%. Hal ini mungkin dikarenakan pada saat proses pengambilan sampel yang kurang sempurna pada wayang Yudhistira. |
| 4 | *Wayang Image Classification using MLP Method and GLCM Feature Extraction* | Untuk memudahkan masyarakat (awam) yang ingin mengetahui wayang dan jenisnya dengan instan. | MLP *(Multi-layer Perceptron)* dan GLCM | Hasil penelitian menunjukkan bahwa klasifikasi wayang menggunakan MLP *(Multi-Layer Perceptron)* metode dan ekstraksi fitur GLCM *(Gray Level Co-Occurrence Matrix)* dapat mengenali wayang objek berdasarkan gambar wayang dan mengklasifikasikannya tidak cukup akurat dan total maksimum akurasi adalah 73,4%. |
| 5 | Penerapan Algoritma *Convolutional Neural Network* Untuk Klasifikasi Motif Citra Batik Solo (Bowo et al., 2020) | Mencoba merancangi sebuah sistem pengklasifikasian motif citra batik Solo yang menggunkan algoritma *Convolutional Neural Network* (CNN) sebagai metode pengenalan pola khususnya klasifikasi gambar batik. | CNN | Model CNN pada penelitian ini menggunakan *input* *shape* berukuran 32x32x3, ukuran filter 3x3, jumlah *epoch* sebanyak 100. Data yang digunakan untuk proses *training* model sebanyak 2256 menghasilkan tingkat akurasi *training* dan testing dalam melakukan deteksi gambar citra batik solo sebesar 99% untuk *accuracy* dan 94% untuk validasi *accuracy* Penelitian ini menggunakan data testing baru sebanyak 745 gambar dimana perkelas terdapat 96 sampai 127 gambar untuk diuji kedalam model yang telah dibuat. Hasil testing menghasilkan tingkat akurasi baru dalam mengklasifikasi motif citra batik solo yaitu sebesar 95%. |
| 6 | *Batik Classification Using Deep Convolutional Network Transfer Learning* (Gultom et al., 2018) | Melakukan penelitian berkelanjutan terkait teknologi pengolahan citra digital yang mampu untuk mengklasifikasi jenis batik berdasarkan polanya. | *Deep-Convolutional Network Transfer Learning* dan Model VGG16 | Ekstraksi fitur otomatis menggunakan *convolutional* pre-trained (Model VGG16) mampu menangani fitur *invarian* transformasi seperti motif Batik sebagus SIFT dan SURF. Akurasinya di atas 80%. |
| 7 | Deteksi dan Pengenalan Ikan Menggunakan Algoritma *Convolutional Neural Network* (Mehindra Prasmatio et al., 2020) | Merancang sebuah sistem yang dapat membantu tugas seorang pakar dalam mengidentifikasi jenis ikan secara otomatis | CNN, Model LeNet5, dan OpenCV | Berdasarkan pengujian dan analisa hasil pengujian dari keseluruhan sistem dan dengan label kelas 9 (jenis ikan), 720 data latih, 180 data uji yang telah dilakukan didapatkan kesimpulan bahwa Akurasi dalam penelitian ini mendapatkan nilai sebesar 85,14% dari hasil 23 dapat memprediksi dengan benar dan 4 tidak dapat diprediksi dengan benar. Selanjutnya mendapatkan nilai presisi sebesar 77,8% dan nilai recall sebesar 85,2%. |
| 8 | Klasifikasi Citra Buah Menggunakan *Convolutional Neural Network* (Fitra Maulana & Rochmawati, 2019) | Menerapkan algoritma CNN (Tensorflow) dan mengklasifikasi citra buah, serta mencari hasil akurasi yang terbaik. Dengan proses pengolahan citra digital dengan CNN, diharapkan dapat membantu para peneliti di bidang perkebunan dan pertanian, botanist, dokter, maupun sebagai media pembelajaran. | CNN-Tensorflow | Hasil dari penelitian yang telah dilakukan yaitu model CNN yang menggunakan perpaduan 3 covolutional layer dan 2 hidden layer mampu mengklasifikasi citra buah-buahan dengan akurasi yang baik. Akurasi yang didapatkan dari proses testing yang menggunakan 345 citra uji menunjukkan angka 97,97%. Model CNN yang dibuat pada penelitian ini juga dapat mengklasifikasi citra buah yang diambil menggunakan kamera *smartphone*. |
| 9 | Klasifikasi Citra Genus Panthera Menggunakan Metode *Convolutional Neural Network* (CNN) (Anwar & Riminarsih, 2019) | Mengklasifikasicitra genus panthera (spesies: harimau, singa, jaguar, macan tutul), karena sebelumnya sudah pernah juga dilakukan penelitian dan mengujian terkait kasus ini namun dengan algoritma Naïve Bayes. | CNN-Tensorflow | Pelatihan model CNN dilakukan dengan steps sebanyak 100, epoch sebanyak 100, batch sebanyak 64, waktu training 46 menit, dataset training menghasilkan akurasi sebesar 0.9231 dan loss 0.2062. Dataset validation memiliki akurasi sebesar 0.8188 dan loss 0.6469. Berdasarkan hasil pengujian maka diperoleh hasil bahwa tingkat akurasi untuk model CNN yang telah dibuat untuk klasifikasi genus panthera adalah sebesar 68%. Meskipun tingkat akurasi belum mencapai 90% tetapi sudah berhasil memperbaiki hasil penelitian sebelumnya terutama untuk klasifikasi macan tutul. Pada penelitian sebelumnya [1] macan tutul belum berhasil diklasifikasikan. Pada penelitian ini, dari 200 citra macan tutul, sudah 49 citra yang berhasil dikenali sebagai macan tutul sedangkan sebagian besar macan tutul dikenali sebagai jaguar. |
| 10 | Klasifikasi Citra Bunga dengan Menggunakan *Deep Learning: CNN (Convolution Neural Network)* (Teknik Elektro & Wega Intyanto, 2021) | Melakukan penelitian klasifikasi citra bunga dengan membuat model arsitektur deep learning, yaitu arsitektur pada CNN yang memiliki kelebihan dalam hal ekstraksi fitur secara otomatis, dengan dua jenis arsitektur CNN: Custom oleh penulis dan memanfaatkan arsitektur VGG16. | CNN-Tensorflow dan Model VGG16 | Hasil training dan uji coba (testing) pada dataset citra jenis bunga dengan arsitektur pada CNN diproses iterasi (*epoch*) sebanyak 50 kali. Pada arsitektur pertama yang di rancangan oleh penulis menunjukkan hasil akurasi yaitu 0.62 dan nilai *loss* yaitu 2.5 (*overfitting*), dilihat dari kurva performa pelatihan lebih baik dibandingkan dengan kurva performa validasi, seta model kurva yang sangat rentang. Hasil yang ditunjukkan pada arsitektur II (VGG16) nilai akurasi yaitu 0.8 dan nilai loss yaitu 0.52. Pada grafik akurasi dapat dilihat bahwa arsitektur II dengan menggunakan arsitektur VGG16 menunjukkan telah mempelajari *dataset* dengan cukup baik dan sebanding pada kedua hasil data (data uji dan testing/validasi). |
| 11 | Analisis dan Implementasi Diagnosis Penyakit Sawit dengan Metode *Convolutional Neural Network* (CNN) (Rasywir et al., 2020) | Merumuskan bagaimana menerapkan metode *Convolutional Neural Network* (CNN) untuk Sistem Diagnosis Penyakit Tanaman Sawit. | CNN-Tensorflow | Dari pengujian sebanyak 2490 citra kelapa sawit berlabel penyakit 11 kategori. Diperoleh hasil akurasi tertinggi sebesar 0.89 dan terendah 0.83 serta rata-rata dengan akurasi 0.87. Hal ini menunjukan bahwa hasil klasifikasi citra kelapa sawit dengan CNN cukup baik. Hasil ini dapat menjadi indikasi pengembangan sistem klasifikasi penyakit sawit otomatis dan *mobile* untuk membantu para petani. |
| 12 | Penentuan Emosi pada Video dengan Convolutional Neural Network (Prasetyawan & Informatika UIN Sunan Kalijaga Yogyakarta Jl Marsda Adi Sucipto Yogyakarta, 2020) | Melakukan penentuan emosi dengan melakukan pengenalan ekspresi wajah manusia dan melakukan perekaman untuk setiap perubahan ekspresi wajah tersebut. | CNN-Tensorflow | Model klasifikasi menggunakan Convolution Neural Network (CNN) menghasilkan akurasi sebesar 74,17%, presisi sebesar 74,07%, dan rata-rata *recall* 74,18%. Pemerataan distribusi data latih dapat meningkatkan kinerja model. Artikel ini menghasilkan sistem klasifikasi yang mampu mengenali setiap perubahan ekspresi wajah kemudian menghitung jumlah setiap jenis ekspresi yang dapat digunakan untuk menentukan emosi seseorang. |

# METODOLOGI PENELITIAN

## Data Penelitian

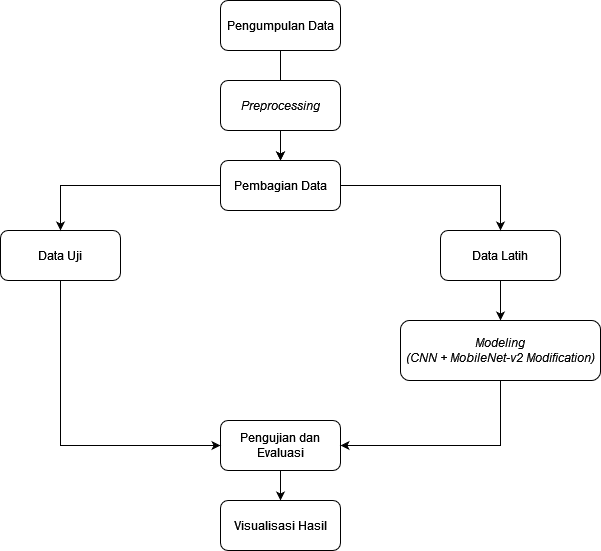
*Dataset* yang digunakan dalam penelitian ini bersumber dari foto-foto wayang kulit di Google dan akun-akun Instagram pengrajin wayang kulit. Jumlah karakter yang direpresentasikan dalam folder sebanyak 22, dengan jumlah foto (wayang kulit) 411 buah, berekstensi *.jpg*. Sedangkan untuk data uji video bersumber dari YouTube *(Public Dataset)*. Berikut adalah contoh cuplikan dataset sebelum dilakukan *Preprocessing* dan *Augmentation*, dapat dilihat pada Gambar 3.1.

**Tabel 3.1 Sampel *Dataset (Images)* Wayang Kulit**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | ***Folder Name*** | **Keterangan** | ***Image*** |
| 1 | Abimanyu | *Raw* |  |
| 2 | Baladewa | *Raw* |  |
| 3 | Buta | *Raw* |  |
| 4 | Kresna | *Raw* |  |
| 5 | Puntadewa | *Raw* |  |

## Penerapan Metode

Untuk membangun sistem aplikasi identifikasi karakter wayang dengan metode *Canny Edge Detection* dan algoritma *CNN* (*custom model* CNN), terdapat beberapa tahapan yang menjadi rancangan utama. Rancangan ini sebagai gambaran proses atau alur aplikasi dari awal sampai dengan akhir, yang terdapat pada Gambar 3.1 berikut.



**Gambar 3.1 Tahapan Metode**

Pada Gambar 3.1 merupakan proses atau alur identifikasi karakter wayang kulit berdasarkan citra dan video. Tahap yang pertama yaitu pengumpulan *Dataset*, yang berupa kumpulan citra wayang kulit dengan 22 jenis atau karakter. *Dataset* tersebut kemudiandiaugmentasi dan *preprocessing* menggunakan metode *Canny Edge Detection* (deteksi tepi). *Output* dari proses *Preprocessing* ini yaitu citra wayang kulit yang sudah dideteksi tepi berukuran piksel *224x224 (.jpg)*. Selanjutnya, kumpulan citra wayang kulit ini akan dibagi menjadi dua bagian, yaitu *Data Train* dan *Data Test* dengan rasio 2:8. *Fix Dataset* inilah yang selanjutnya disimpan ke *Google Drive* sebagai *database* saat dilakukan proses *build-up* model CNN menggunakan *custom* model CNN dengan *base model MobileNet-v2.* Setelah dilakukan proses *training model* dengan *ephoch*s sebanyak 50 kali, dan diperoleh nilai akurasi yang baik, maka *custom* model CNN akan diimplementasikan pada sistem aplikasi untuk keperluan identifikasi atau klasifikasi karakter wayang kulit. Hasil akhir dari proses ini berupa visualisasi hasil prediksi nama karakter wayang kulit.

### Pengumpulan Data

*Dataset* ini berupa kumpulan citra wayang kulit dengan jumlah karakter sebanyak 22. Citra wayang kulit tersebut diperoleh dari *Google* dan akun-akun *Instagram* pengrajin wayang kulit. Adapun nama-nama karakter wayang kulit yang digunakan sebagai dataset ini merupakan karakter tokoh wayang kulit dari cerita *Mahabharata*, antara lain: *abimanyu, anoman, arjuna, bagong, baladewa, bima, buta, cakil, durna, dursasana, duryudana, gareng, gatotkaca, karna, kresna, nakula\_sadewa, patih\_sabrang, petruk, puntadewa, semar, sengkuni,* dan *togog.* Jumlah file citra wayang kulit dari tahap pengumpulan data ini berjumlah 411 dengan beragam ukuran dimensi, berekstensi *.jpg.* Kumpulan citra tersebut kemudian akan diolah agar lebih variative (dengan *data augmentation*) dan *preprocessing*.

### *Preprocessing*

Pada tahap *preprocessing* ini, dilakukan beberapa proses untuk menambah variasi *dataset* dengan teknik *data* *augmentation*, dan deteksi tepi. *Augmentation* atau augmentasi data adalah proses untuk memodifikasi dataset yang sudah ada agar lebih variatif. dan Berikut adalah rincian dari tahap *preprocessing* citra wayang kulit.

*Flip Left-Right*

Flip Left-Right merupakan proses membalik suatu citra kanan-kiri atau sebaliknya. Presentase proses *flip* ini yaitu 100%, sehingga jika dalam sebuah folder terdapat 5 buah citra, maka kelimanya akan di-*flip*.

*Flip Up-Down*

*Flip Up-Down* merupakan proses membalik suatu citra atas-bawah atau sebaliknya. Presentase proses *flip* ini yaitu 100%, sehingga jika dalam sebuah folder terdapat 5 buah citra, maka kelimanya akan di-*flip*.

*Rotation*

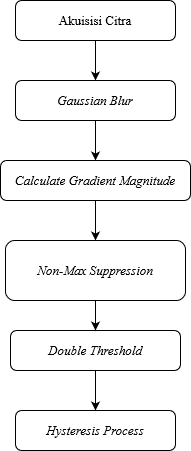
*Rotation* atau rotasi merupakan proses memutar suatu citra dengan sudut tertentu. Dalam penelitian ini, besar sudutnya adalah 30 derajat.

*Zooming*

*Zooming* atau *Scaling* merupakan proses memperbesar atau memperkecil suatu citra. Dalam penelitian ini dilakukan proses *zooming* dengan presentase 40%, 100%, 80%, dan 120% pada setiap sumbunya.

*Edge Detection Process*

*Edge Detection* atau deteksi tepi adalah teknik pemrosesan citra, yang digunakan untuk mengidentifikasi batas (tepi) objek, atau wilayah dalam suatu citra. Dalam penelitian ini fungsi deteksi tepi yang digunakan yaitu *Canny* dari *library* OpenCV.



**Gambar 3.2 Alur Metode *Canny Edge Detection***

Pada Gambar 3.2 merupakan alur dari metode canny edge detection *(scratch).* Terdapat enam (6) langkah pada metode ini, antara lain: akuisisi citra, *gaussian blur, calculate gradient magnitude, non-max suppression, double threshold,* dan *hysteresis process*. Langkah pertama adalah akuisisi citra, yaitu proses untuk membaca citra yang akan diolah, dan dimasukkan ke dalam sebuah variabel. Selanjutnya, citra akan masuk ke tahap *noise reduction*, yaitu proses menghaluskan citra dengan *GaussianBlur (5x5)*. Setelah itu, citra akan diproses dengan *sobel kernel* dengan menerapkan *gradient magnitude*. Citra hasil proses ini, sudah terlihat tepiannya, namun masih perlu ditipiskan untuk menyeleksi piksel tepi atau bukan dengan operasi *non-maximum suppression.* Selanjutnya adalah menentukan nilai *threshold* bawah dan *threshold* atas, dan kedua nilai tersebut akan digunakan pada tahap terakhir, yaitu *hysteresis process.*

### Pembagian Data

Sebelum proses pembagian data, citra wayang kulit hasil preprocessing akan dikelompokkan ke dalam masing-masing folder yang berjumlah 22, sesuai nama karakter wayang kuit tersebut. Setelah pengelompokkan data, dataset akan dibagi menjadi dua (2) bagian, yaitu: data uji dan data latih, dengan rasio perbandingan 2:8. Ilustrasi pembagian dataset disajikan pada Gambar 3.1 berikut.

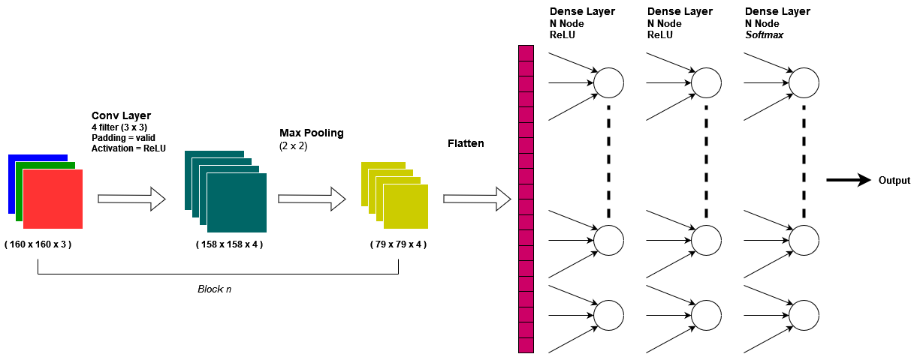


**Gambar 3.2 Pembagian *Dataset***

Pada masing-masing data latih dan data uji, terdapat 22 folder yang merupakan label karakter wayang kulit. Setiap folder, berisi citra wayang kulit hasil augmentasi dan *preprocessing*.

### *Modeling*

Tahap *modeling* dilakukan untuk mengekstraksi fitur dari citra wayang kulit hasil preprocessing menggunakan arsitektur atau algoritma *Convolutional Neural Network (CNN)*. Pada arsitektur CNN, terdapat beberapa proses utama yang dikenal dengan istilah layer, antara lain: *Convolutional Layer, Pooling Layer, Flatten, Dense Layer, Activation*. Pada Gambar 3.3 disajikan ilustrasi arsitektur CNN seperti berikut.



**Gambar 3.3 Arsitektur CNN**

Langkah pertama pada pengolahan citra menggunakan algoritma CNN yaitu operasi konvolusi. Citra merupakan sebuah matriks yang beisi nilai piksel. Sehingga, pada layer pertama *(convolution layer)* ini dilakukan proses konvolusi antara matriks dari citra masukan (input) dengan kernel atau filter dengan ordo matriks dan nilai matriks tertentu. Pada Gambar 3.3 diilustrasikan dimensi dari citra masukan yaitu (160 x 160 x 3). Hal itu berarti, citra X mempunyai *160 piksel baris x 160 piksel kolom x 3 channel* (RGB atau citra berwarna). Matriks tersebut dikonvolusi dengan 4 buah kernel atau filter dengan ukuran 3x3. Hasil dari proses konvolusi citra disebut *feature map* sebanyak empat (4) buah (karena jumlah kernelnya 4). Perhitungan operasi konvolusi disajikan pada persamaan 3. 1 berikut.

|  |  |
| --- | --- |
|  | 3. 1 |

Keterangan:

*m =* piksel baris kolom

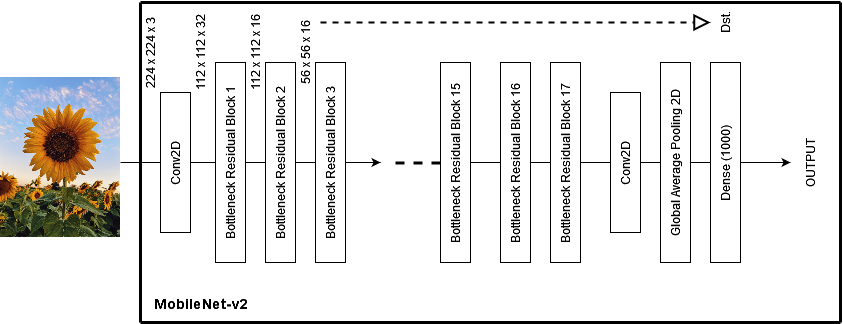
*n =* piksel pada kolom

*f =* ordo matriks kernel atau filter

Tahap selanjutnya yaitu *pooling* atau *subsampling*, berfungsi untuk mengurangi dimensi dari setiap feature map hasil operasi konvolusi. Beberapa jenis dari *pooling layer* antara lain: *MaxPooling, AvgPooling, dan SumPooling.*Pada MaxPooling (misal: kernel 2x2) berarti setiap elemen matriks 2x2 pertama pada *feature map* hasil konvolusi, diambil satu nilai terbesar. Proses tersebut berjalan ke seluruh elemen feature map (hasil konvolusi). Sederhanaya, jika terdapat feature map berdimensi (158 x 158 x 3), hasil dari operasi MaxPooling berupakan satu per dua (setengah) dari matriks feature map, berarti menjadi (79 x 79 x 3). Pada tahap ini biasanya dilakukan proses aktivasi. Terdapat beberapa jenis metode aktivasi, dan yang paling sering digunakan yaitu ReLU *(Rectified Linear Unit)*. *ReLU activation* banyak digunakan karena mempercepat proses training, disbanding dengan fungsi aktivasi yang lain (sigmoid, tanH, linear, dll) (Primartha, 2021). Proses konvolusi, pooling, dan aktivasi dapat dilakukan berulang-ulang pada penerapan algoritma CNN, tujuannya adalah untuk memperbanyak proses ekstraksi sehingga algoritma CNN akan semakin “cerdas”.

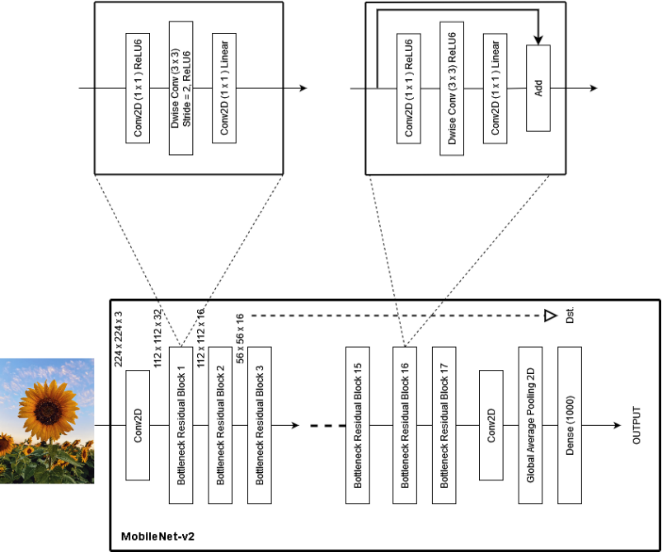
Langkah berikutnya yaitu *flatten* atau *flattening*, adalah proses mengubah feature map menjadi array satu (1) dimensi yang kemudian akan dimasukkan ke lapisan *Neural Network (Dense).* Selanjutnya array tersebut akan menjadi input bagi lapisan neural network seperti pada sub sub bab 2.7.2. Dan pada layer terakhir merupakan *Dense* atau *Fully Connected Layer* dengan angka yang merepresentasikan jumlah label atau kelas. Pada kasus yang hanya memiliki keluaran 1 node, maka akan digunakan fungsi aktivasi sigmoid. Sedangkan pada penelitian ini, karena jumlah label lebih dari satu, maka digunakan fungsi aktivasi *softmax* yang sangat cocok dengan *multiclass classification.*

Selain arsitektur CNN dasar seperti penjelasan di atas, terdapat beberapa jenis model latih yang merupakan model pra-latih atau *pre-trained model*. Berikut beberapa jenis model pra-latih antara lain: MobileNet-v1, MobileNet-v2, VGG16, ResNet, LeNet, dll. Pada penelitian ini, penulis menerapkan MobileNet-v2 sebagai base model dengan beberapa modifikasi pada layer-layernya, dan dilakukan training dengan dataset citra wayang kulit, sehingga menghasilkan *custom model CNN*. Berikut pada Gambar 3.4 merupakan ilustrasi layer pada arsitektur MobileNet-v2.



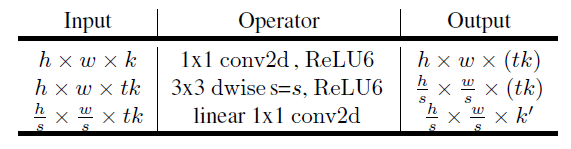
**Gambar 3.4 Arsitektur Default MobileNet-v2**

Pada prinsipnya, MobileNet-v2 pada layer-layernya melakukan operasi yang sama dengan arsitektur CNN dasar. Namun jumlahnya saja yang lebih banyak dan kompleks. Detail ilustrasi arsitektur MobileNet-v2 dapat dilihat pada Gambar 3.5 berikut.



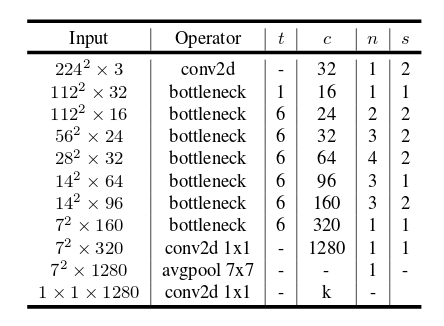
**Gambar 3.5 Detail Layer MobileNet-v2**

Pada arsitektur MobileNet-v2 terdapat dua (2) jenis blok yaitu: residual block (stride = 1) dan blok lainnya dengan (s =2). Pada masing-masing kedua blok tersebut, mempunyai tiga (3) lapisan, antara lain: *1x1 convolution with ReLU6, depthwise convolution,* dan *1x1 convolution without any non-linearitas*. Berikut disajikan detail informasi terkait blok-blok pada arsitektur MobileNet-v2 seperti Gambar 3.6.



**Gambar 3.6 Detail Blok Arsitektur MobileNet-v2**

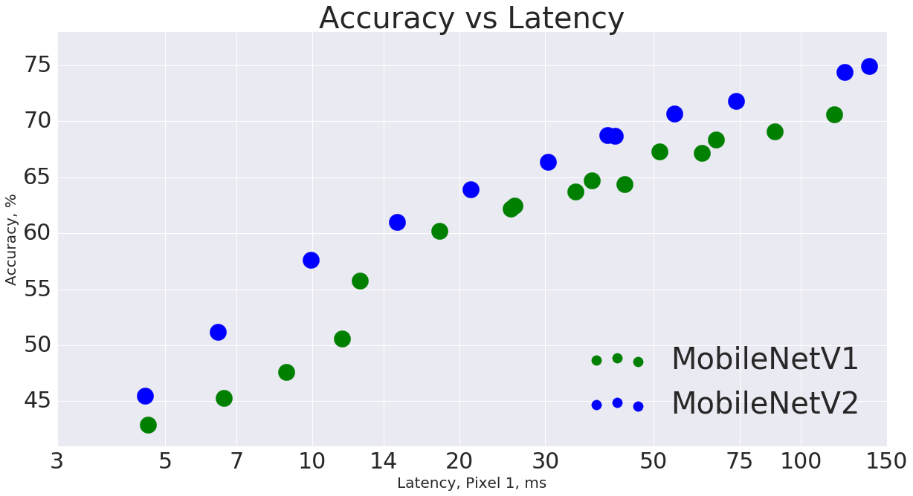
Terdapat faktor ekspansi t. Nilai t=6 (ReLU6) untuk semua percobaan utama. Jika input dengan 64 channel, maka akan memiliki dimensi output 64 x t atau 64 x 6 = 384 channel.



**Gambar 3.7 Layer-Layer Arsitektur MobileNet-v2**

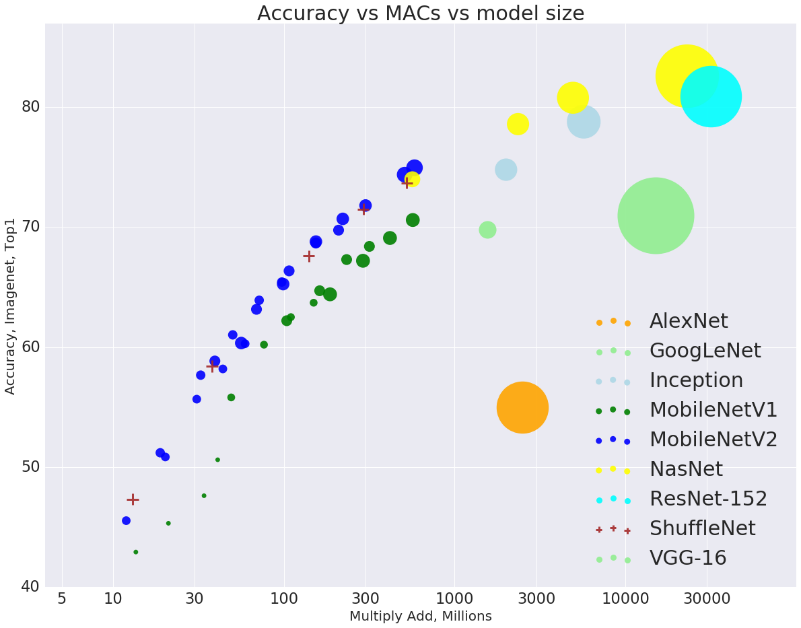
Pada Gambar 3.7 bagian *bottleneck* terdapat *input* dan *output* antara model sedangkan lapisan atau *layer* bagian dalam mengenkapsulasi kemampuan model untuk mengubah *input* dari konsep tingkat yang lebih rendah (i.e. piksel) ke deskriptor tingkat yang lebih tinggi (i.e. kategori gambar). Pada akhirnya, seperti halnya koneksi residual pada CNN tradisional, *shortcut* antar *bottlenecks* memungkinkan training atau pelatihan yang lebih cepat dan akurasi yang lebih baik.

Berikut adalah perbandingan waktu ekseskusi antara MobileNetV1 dan MobileNetV2 menggunakan TF-Lite pada large core dari Piksel 1 ponsel. Dapat dilihat pada Gambar 3.7 bahwa MobileNetV2 meningkatkan kecepatan (mengurasi latensi) dari MobileNetV1.



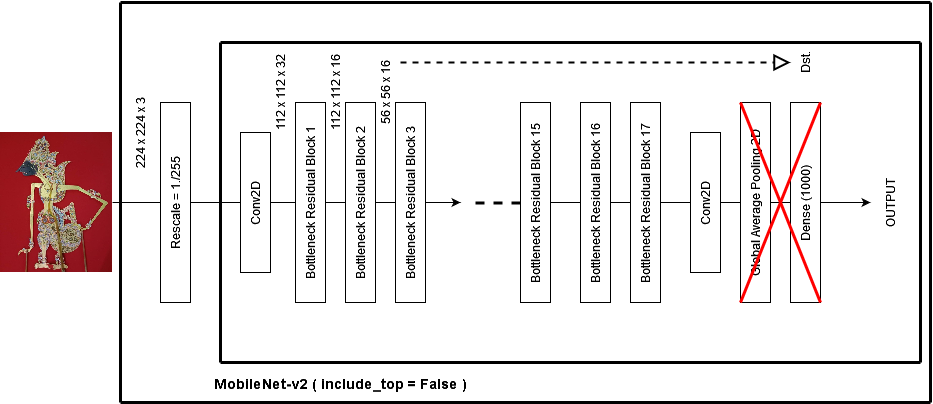
**Gambar 3.7 Perbandingan Performa MobileNet-v1 dan MobileNet-v2**

Berikut perbandingan dari berbagai metode dari segi MAC — Multiply Accumulates, metode evaluasi dengan menghitung jumlah multiply-accumulate yang dibutuhkan untuk komputasi pada satu gambar, disajikan pada Gambar 3.8.



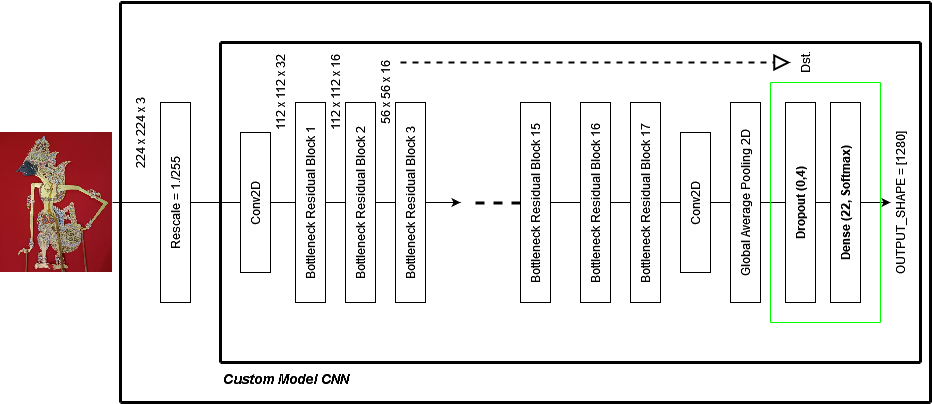
**Gambar 3.8 Perbandingan MobileNet-v2 dengan Pre-trained Model**

Berdasarkan uraian tentang arsitektur CNN dan MobileNet-v2 tersebut, maka dalam penelitian ini penulis menggunakan kombinasi antara keduanya. Modifikasi dari model MobileNet-v2 dilakukan karena tidak sesuai dengan kebutuhan pada rumusan masalah penelitian ini, yaitu mengklasifikasikan 22 nama karakter wayang. Sedangkan pada *default* MobileNet-v2, node pada layer terakhir berjumlah 1000 (diperuntukkan untuk mengklasifikasikan 1000 jenis citra). Sehingga, perlu adanya modifikasi dengan menghapus beberapa layer dan menambahkannya sesuai kebutuhan. Ilustrasi tentang hal ini disajikan pada Gambar 3.9.

**No table of figures entries found.**

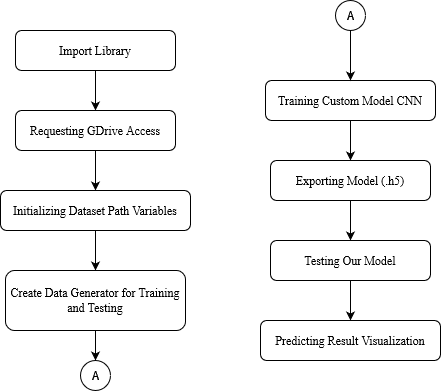
**Gambar 3.9 Remove Layer MobileNet-v2**

Pada Gambar 3.9, dua layer terakhir pada arsitektur default MobileNet-v2 dihapus, dengan perintah *include\_top = False*. Dengan hal ini, maka layer tambahan sesuai konsep CNN dasar bisa ditambahkan. Pada penelitian ini, penulis menambahkan ­*Rescaling, Dropout,* dan *Dense (22)* seperti Gambar 3.10 berikut.



**Gambar 3.7 Custom Model CNN**

Maka, proses modeling pada penelitian ini menggunakan custom model CNN dengan base model MobileNet-v2. Pada Gambar 3.11 disajikan proses modeling dengan yang di-run pada *Google Colaboratory* seperti berikut.

****

**Gambar 3.11 Proses Training Custom Model CNN**

Beberapa hal yang perlu diperhatikan pada proses *training custom* model ini adalah jumlah data latih dan data uji (sesuai sub sub bab 3.2.3), *epochs* (iterasi), dan nilai *loss validation-accuracy validation.* Model dengan akurasi baik akan diekspor dengan ekstensi file *.h5*.

### Identifikasi Data Uji

Pada tahap identifikasi data uji dilakukan pengujian terhadap *custom* model CNN (kombinasi *base model* MobileNet-v2 dan layer CNN) yang sudah diimplementasikan pada aplikasi. Data uji merupakan citra wayang kulit yang masih mentah (belum dilakukan *preprocessing*). Hasil dari identifikasi ini berupa nama karakter wayang dengan presentase kemiripannya. Pada penelitian ini, *output* dari identifikasi termasuk kategori *multiclass classification,* karena mempunyai label lebih dari dua*.* Proses identifikasi data uji akan dijabarkan pada sub sub bab 4.3.6.

### Visualisasi Hasil

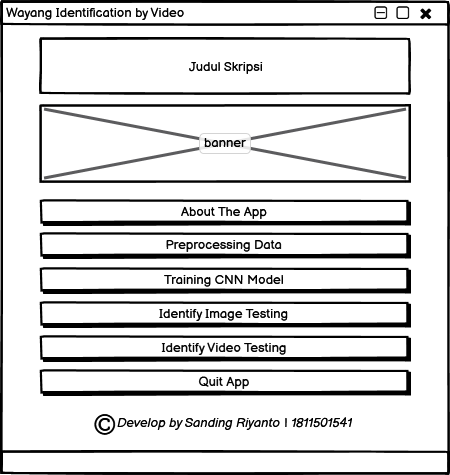
Visualisasi hasil identifikasi data uji ditampilkan pada sebuah page atau frame. Dalam page tersebut akan menampilkan *text* atau label karakter wayang kulit hasil prediksi (warna biru: benar, merah: salah). Selain itu, data uji (citra wayang kulit) juga akan ditampilkan beserta *messagebox* berisi informasi terkait hasil identifikasi.

## Rancangan Layar Aplikasi

Rancangan layar adalah langkah-langkah atau urutan dalam membuat tampilan sebelum digunakan dalam sebuah sistem aplikasi dengan tujuan perkiraan yang baik agar tampilan dapat digunakan oleh pengguna (Mufti, 2015). Berikut adalah rancangan layar dari sistem aplikasi pendeteksi nama karakter wayang kulit.

### Rancangan Layar Menu Utama

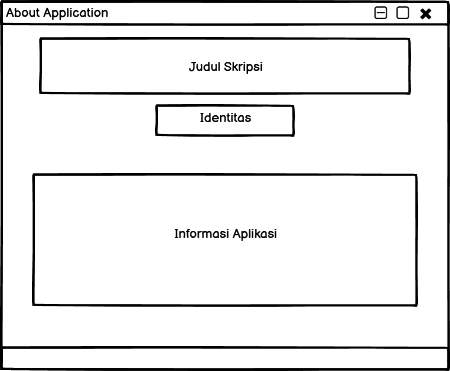
Rancangan layar Menu Utama adalah rancangan tampilan aplikasi ketika program pertama kali dijalankan. Pada rancangan layer ini terdapat beberapa bagian, yaitu: *header* (judul), *banner*, dan beberapa *menu button* seperti pada Gambar 3.6.



**Gambar 3.6 Rancangan Layar Menu Utama**

### Rancangan Layar *About Application*

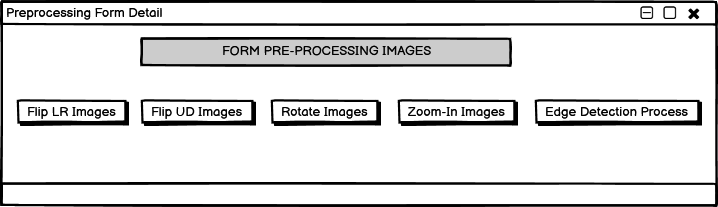
Rancangan layar *about application* merupakan rancangan tampilan aplikasi yang berisi detail informasi tentang system aplikasi pendeteksi nama karakter wayang kulit. Detail informasi ditampilkan *by text* seperti pada Gambar 3.7.



**Gambar 3.7 Rancangan Layar *About Application***

### Rancangan Layar *Preprocessing Images*

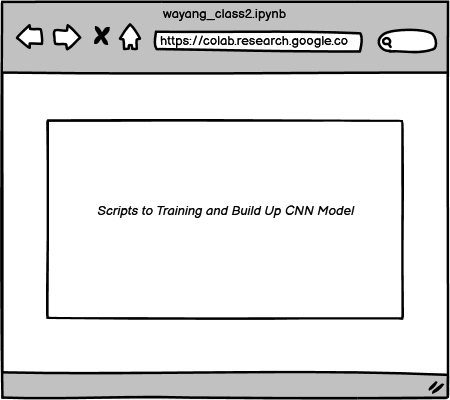
Rancangan layar *preprocessing images* adalah rancangan tampilan aplikasi yang berisi berbagai sub-menu (dari menu *preprocessing data)*, antara lain: *Flip-LR, Flip-UD, Rotation, Zooming,* dan *Edge Detection Process*. Rancangan layar ini disajikan pada Gambar 3.8.



**Gambar 3.8 Rancangan Layar *Preprocessing Images***

### Rancangan Layar *Training CNN Model*

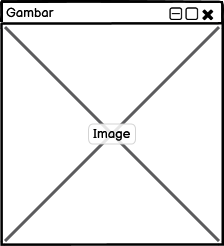
Pada Gambar 3.9 merupakan rancangan layar saat training *CNN Model*, di *Google Colaboratory*. Sehingga, rancangan layar ini adalah tampilan *Google Colaboratory*, dengan baris-baris kode untuk proses *training* model.



**Gambar 3.9 Rancangan Layar *Training CNN Model***

### Rancangan Layar *Show Image-Video*

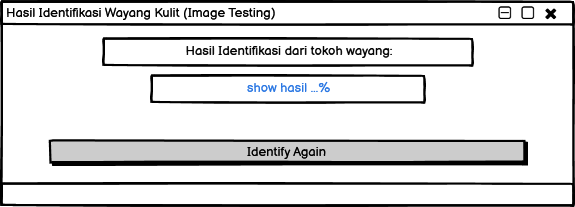
Rancangan layar *show image-video* adalah rancangan tampilan ketika system aplikasi menampilkan gambar atau video (saat *preprocessing data* dan *identify*). Rancangan layar *show image-video* disajikan pada Gambar 3.10 berikut.



**Gambar 3.10 Rancangan Layar *Show Image-Video***

### Rancangan Layar *Identify Image*

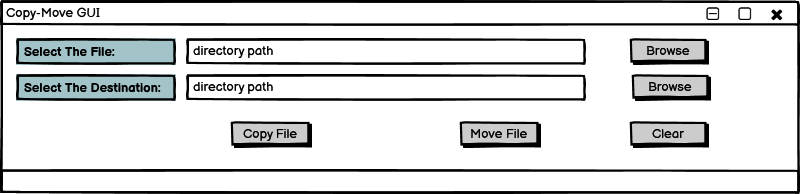
Pada Gambar 3.11 merupakan rancangan layar *identify image* (sub-menu *identify image*). Pada page berisi visualisasi hasil identifikasi nama karakter wayang kulit dari data *input* gambar.



**Gambar 3.11 Rancangan Layar *Identify Image***

### Rancangan Layar *Split Dataset*

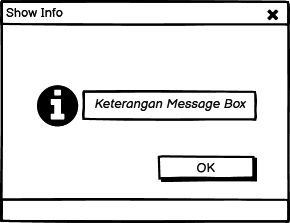
Pada Gambar 3.13 merupakan rancangan layar *split dataset*. Halaman ini sifatnya *portable*, dan hanya berfungsi sebagai user interface ketika pembagian *dataset* menjadi *data training* dan *data testing.*



**Gambar 3.13 Rancangan Layar *Split Dataset***

### Rancangan Layar *Message Box*

Pada Gambar 3.14 merupakan rancangan layar *message box*. Message box ini muncul sebagai notifikasi ketika tahap *prepocessing* *data* dan identifikasi data *(testing).*



**Gambar 3.14 Rancangan Layar *Message Box***

## Rancangan Pengujian

Pengujian dilakukan untuk mengetahui nilai atau tingkat akurasi, presisi, dan *recall* dari model latih menggunakan algoritma yang diusulkan. Pada penelitian ini, pengujian dilakukan dengan cara membandingkan beberapa data hasil prediksi (data hasil tahap klasifikasi) dengan sekumpulan data aktual (data hasil tahap *labeling*). Adapun dimaksud dengan beberapa data hasil prediksi yaitu sekumpulan data yang telah diproses melalui algoritma CNN *(Pre-trained Model)*.

### Akurasi

Akurasi merupakan tingkat kedekatan antara nilai prediksi dengan nilai aktual persamaan (3. 1).

### Presisi

Presisi merupakan tingkat ketepatan antara informasi yang diminta dengan jawaban yang diberikan oleh sistem persamaan (3.2).

### *Recall*

*Recall* merupakan tingkat keberhasilan sistem dalam menemukan kembali sebuah informasi persamaan (3. 3).

Pada penelitian ini, pengukuran tingkat akurasi, presisi, dan *recall* dapat diketahui melalui *confusion matrix* dengan persamaan sebagai berikut:

|  |  |
| --- | --- |
|  | (3. 2) |
|  | (3. 3) |
|  | (3. 4) |

Keterangan:

1. *True Positive* (TP) merupakan data positif yang diprediksi benar. Misalnya: *tweet* 1 ber*label* positif dan dari *model* latih yang dibuat memprediksi *tweet* 1 bernilai positif juga.
2. *True Negative* (TN) merupakan data negatif yang diprediksi benar. Misalnya: *tweet* 1 ber*label* negatif dan dari *model* latih yang dibuat memprediksi *tweet* 1 bernilai negatif juga.
3. *False Positive* (FP) merupakan data negatif namun diprediksi sebagai data positif. Misalnya: *tweet* 1 ber*label* negatif namun dari *model* latih yang dibuat memprediksi *tweet* 1 bernilai positif.
4. *False Negative* (FN) merupakan data positif namun diprediksi sebagai data negatif. Misalnya: *tweet* 1 ber*label* positif namun dari *model* latih yang dibuat memprediksi *tweet* 1 bernilai negatif.

# HASIL DAN PEMBAHASAN

## Lingkungan Percobaan

Agar aplikasi yang telah dikembangkan dapat berjalan dengan semestinya, dibutuhkan perangkat dengan spesifikasi tertentu. Adapun dalam penelitian ini menggunakan spesifikasi perangkat diantaranya.

### Spesifikasi perangkat keras

Daftar perangkat keras yang mendukung aplikasi ini untuk berjalan dengan baik adalah sebagai berikut:

Processor : AMD Ryzen™ 3 / 5 Processor

RAM : 8,00 GB

SSD : 512 GB

VGA : AMD Radeon Vega 8 Graphics

### Spesifikasi perangkat lunak

Daftar perangkat keras yang mendukung aplikasi ini untuk berjalan dengan baik adalah sebagai berikut:

Sistem Operasi : Windows 10 Home

Bahasa Pemrograman : Python 3.9 (64 bit)

IDE : Visual Studio Code v1.68.0

Browser : Google Chrome

## Implementasi Metode

Implementasi metode dalam penelitian ini dilakukan dengan enam (6) tahapan utama (mengacu dari sub-bab 3.2). Adapun alur tahapan implementasi metode yang dimaksud antara lain: tahap pengumpulan data, tahap *preprocessing*, tahap pembagian data *(splitting)*, tahap training model CNN, tahap klasifikasi *(testing)*, dan visualisasi hasil.

### Tahap Pengumpulan Data

Berdasarkan sub bab (3. 1), dataset pada penelitian ini berupa kumpulan citra wayang kulit. Dataset tersebut diperoleh dari Google dan Instagram (akun-akun pengrajin wayang kulit) dimulai pada 18 April hingga 18 Mei 2022. Total raw dataset berjumlah 411 buah dengan format ekstensi file *.jpg*. Sedangkan total fix dataset (setelah di-*prepocessing*) bertambah menjadi 6602 buah, berekstensi *.jpg* dan berukuran 224x224 piksel. Fix dataset tersebut kemudian dimasukkan ke dalam folder (22 buah) yang diberi nama sesuai dengan label data (nama-nama karakter wayang kulit).

### Tahap *Preprocessing*

Tahap *preprocessing* merupakan proses untuk menyiapkan dataset sebelum masuk ke tahap berikutnya. Pada penelitian ini terdapat lima (5) tahap preprocessing data, antara lain: *flip left-right, flip up-down, rotation, zooming,* dan *edge detection process.*Berikut adalah penjabaran dari tahap *preprocessing*.

1. *Flip Left-Right*

Pada Tabel 4.1 merupakan cuplikan dataset setelah diproses dengan metode Flip Left-Right. Citra wayang kulit yang sebelumnya menghadap ke kiri dibalik menjadi menghadap ke kanan, begitu sebaliknya.

**Tabel 4.1 Flip Left-Right**

|  |  |
| --- | --- |
| Keterangan | Gambar |
| Sebelum |  |
|  |
|  |
| Sesudah |  |
|  |
|  |

1. *Flip Up-Down*

Pada Tabel 4.2 merupakan cuplikan dataset setelah diproses dengan metode Flip Up-Down. Citra wayang kulit akan dicerminkan secara vertikal atau atas-bawah.

**Tabel 4.2 Flip Up-Down**

|  |  |
| --- | --- |
| **Keterangan** | **Gambar** |
| Sebelum |  |
|  |
|  |
| Sesudah |  |
|  |
|  |

1. *Rotation*

Pada Tabel 4.3 merupakan cuplikan *dataset* setelah diproses dengan metode *rotation*, dengan besar sudut (-25 derajat, 25 derajat).

**Tabel 4.3 *Rotation***

|  |  |
| --- | --- |
| **Keterangan** | **Gambar** |
| Sebelum |  |
|  |
|  |
| Sesudah |  |
|  |
|  |

1. *Zooming*

Pada Tabel 4.4 merupakan cuplikan *dataset* setelah diproses dengan metode *zooming* atau *scaling*, dengan presentase 40%, 100%, 80%, dan 120% pada setiap sumbunya.

**Tabel 4.4 *Zooming***

|  |  |
| --- | --- |
| **Keterangan** | **Gambar** |
| Sebelum |  |
|  |
|  |
| Sesudah |  |
|  |
|  |

1. *Edge Detection Process*

*Canny Edge Detection* adalah algoritma deteksi tepi yang populer. Metode ini dikembangkan oleh John F. Canny. Algoritma untuk deteksi tepi menggunakan fungsi *Canny* memiliki detail proses sebagai berikut :

1. *Noise Reduction (Gaussian Blur)*

Langkah pertama pada proses deteksi tepi menggunakan algoritma *Canny* adalah menghilangkan *noise* pada citra, dengan filter *Gaussian* yang berukuran filter matriks 5x5. Semakin besar ukuran *matrix filter,* maka akan semakin blur citra dan *noise*-nya.

1. *Gradient Calculation*

Citra yang telah dihaluskan kemudian disaring dengan kernel *Sobel,* baik dalam arah horizontal maupun vertikal untuk mendapatkan turunan pertama dalam arah horizontal *(Gx)* dan arah vertikal *(Gy)*. Dari kedua parameter tersebut, kemudian dicari gradien tepi, dan arah untuk setiap piksel seperti pada persamaan (4. 1) berikut.

|  |  |
| --- | --- |
|  | (4. 1) |

Sedangkan untuk mencari arah atau sudut yang disimbolkan *theta* disajikan pada persamaan (4. 2) berikut.

|  |  |
| --- | --- |
|  | (4. 2) |

Arah atau sudut gradien selalu tegak lurus terhadap tepi. Kemudian dibulatkan ke salah satu dari empat sudut yang mewakili vertikal, horizontal dan dua arah diagonal.

1. *Non-Maximum Suppression*

Setelah mendapatkan besaran dan arah gradien, pemindaian penuh gambar dilakukan untuk menghilangkan piksel yang tidak diinginkan yang mungkin bukan merupakan tepi. Untuk ini, pada setiap piksel, piksel diperiksa apakah itu maksimum lokal di lingkungannya dalam arah gradien. Atau dengan kata lain, proses *Non-Maximum Suppression* bertujuan untuk mereduksi piksel dengan ke arah piksel dengan nilai maksimum pada arah (sudut) yang sama. Ilustrasi *non-maximum suppression* seperti pada Gambar 4.1 berikut.



**Gambar 4.1 Ilustrasi *Non-Maximum Suppression* (Sumber: https://docs.opencv.org/3.4/da/d22/tutorial\_py\_canny.html)**

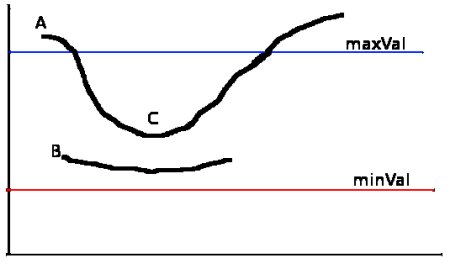
Titik A berada di tepi (dalam arah vertikal). Arah gradien normal ke tepi. Titik B dan C berada dalam arah gradien. Jadi, titik A diperiksa dengan titik B dan C untuk melihat apakah itu membentuk maksimum lokal. Jika demikian, maka dianggap untuk tahap berikutnya, jika tidak, hal itu ditekan (dimasukkan ke nol). Sehingga, citra yang dihasilkan dari proses ini adalah gambar biner dengan tepi yang lebih tipis.

1. *Double Threshold*

Dalam proses ini diminta memasukkan dua (2) parameter yaitu *high threshold* (Th) dan *low threshold* (Tl). Double threshold bertujuan untuk mengidentifikasi tiga (3) jenis piksel, yaitu: kuat, lemah, dan non-relevan. Piksel kuat adalah piksel yang punya intensitas tinggi, dan bisa dipastikan tepi atau *edge*. Piksel lemah yaitu piksel yang mempunyai intensitas tidak cukup tinggi, sehingga belum dapat ditentukan apakah tepi atau bukan. Piksel non-relevan adalah piksel yang dipastikan bukan tepi atau *edge*.

1. *Edge Tracking by Hysteresis*

Tahap terakhir dalam deteksi tepi dengan metode *Canny* yaitu *Hysteresis*. *Edge Tracking by Hysteresis* merupakan proses untuk memastikan apakah piksel-piksel lemah dapat dikategorikan sebagai *edge* atau tidak. Jika di sekitar (tetangga) piksel lemah tidak terdapat piksel kuat *(strong)*, maka piksel tersebut dianggap bukan *edge*. Dan jika di sekitar piksel lemah terdapat piksel kuat *(strong)*, maka piksel tersebut merupakan *edge*.



**Gambar 4.2 *Hysteresis Threshold* (Sumber: https://docs.opencv.org/3.4/da/d22/tutorial\_py\_canny.html)**

Pada Gambar 4.2 merupakan ilustrasi operasi *hysteresis* dengan nilai maksimum *(maxVal)* dan nilai minimum *(minVal)*. Tepi A berada di atas nilai maksimum, sehingga dianggap sebagai *edge* (pasti). Tepi C merupakan tepi dengan piksel lemah, namun karena terhubung dengan tepi A (representasi dari piksel tetangga yang kuat), maka tepi C dianggap sebagai *edge*. Sedangkan tepi B walaupun dapat dikategorikan tepi berpiksel lemah, tetapi tidak terhubung dengan tepi A, atau piksel tepi B tidak mempunyai tetangga dengan piksel kuat, sehingga tepi B bukan ­*edge*.

Pada penelitian ini, digunakan metode *Canny Edge Detection* dari pustaka atau *library* OpenCV. Tetapi, penulis juga mengimplementasikan *Canny Edge Detection* sesuai persamaan (4. 1) dan (4. 2), serta ilustrasi pada Gambar 4.1 dan 4.2.

Pada Tabel 4.5 merupakan cuplikan dataset setelah diproses dengan metode *Canny Edge Detection* dengan *high threshold* (Th=200) dan *low threshold* (Tl=100). Citra wayang kulit hasil pendeteksian tepi ini yang menjadi *fix dataset* untuk *training* model CNN.

**Tabel 4.5 *Edge Detection***

|  |  |
| --- | --- |
| **Keterangan** | **Gambar** |
| Sebelum |  |
|  |
|  |
| Sesudah |  |
|  |
|  |

### Tahap Pembagian Data

Tahap pembagian data merupakan proses membagi *fix dataset* menjadi dua (2) bagian, yaitu: *data training* (data latih) dan *data testing* (data uji). Pembagian data dilakukan sesuai rasio yang ditentukan, yaitu 2:8 (data uji:data latih) atau 20% data uji, dan 80% data latih.

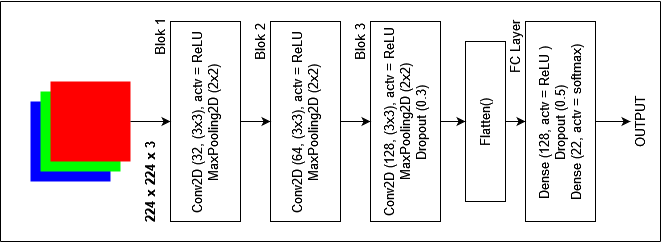
Dalam penelitian ini, tahap pembagian data dilakukan terhadap 6.602 citra wayang kulit berlabel. Dengan rasio data 2:8, maka dapat diketahui jumlah data uji yang diperoleh sebanyak 1.312 citra wayang kulit berlabel, dan jumlah data latih yang diperoleh sebanyak 5.290 citra wayang kulit berlabel. Hasil dari pembagian data tersebut kemudian akan disimpan atau di-*upload* ke *Google Drive (Cloud Data Storage)* untuk diproses pada tahap *training* model CNN.

### Tahap *Modeling*

Berdasarkan sub sub bab 3.2.4, tahap *modeling* pada penelitian ini menggunakan custom model CNN, yang merupakan kombinasi atau modifikasi dari base model MobileNet-v2 dan arsitektur CNN classic. Penulis melakukan percobaan keduanya, sehingga diperoleh hasil seperti berikut.

Arsitektur CNN classic

Pada percobaan pertama, penulis membuat arsitektur CNN dengan komposisi layer seperti pada Gambar 4.1.



**Gambar 4.1 Arsitektur CNN Classic**

Gambar 4.1 merupakan arsitektur CNN classic untuk percobaan modeling ke-1. Terdapat tiga (3) blok yang berisi layer konvolusi dan pooling, serta dropout layer pada blok ke-3. Kemudian masuk flatten layer untuk dijadikan array satu dimensi, yang selanjutnya akan masuk fully connected layer, dan output layer.

Sesuai persamaan (3. 1) citra yang diproses pada layer konvolusi dengan filter atau kernel tertentu akan menghasilkan feature map dengan dimensi yang lebih kecil dan banyaknya sesuai jumlah kernel. Pada Gambar 4.1, citra awal memiliki dimensi *(m x n x channel)* = 224 x 224 x 3 (RGB). Maka, berikut disajikan perhitungan konvolusi citra dengan yang berjumlah 32.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

Selanjutnya, *feature map* masuk pooling layer dengan jenis *MaxPooling*, kernel matriks (2 x 2). Sehingga dimensi citra hasil MaxPooling ini adalah satu per dua dari feature map hasil operasi konvolusi. Berarti hasil pooling layer menghasilkan citra berdimensi *111 x 111 x 32.*

Pada blok 2 dilakukan operasi konvolusi dan pooling seperti pada blok 1. Kali ini jumlah kernel sebanyak 64 dengan matriks ordo (3 x 3). Berikut adalah perhitungannya:

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

Berikutnya, feature map tersebut akan dilakukan pooling sama seperti pada blok 1. Sehingga, menghasilkan citra berdimensi *54 x 54 x 64*.

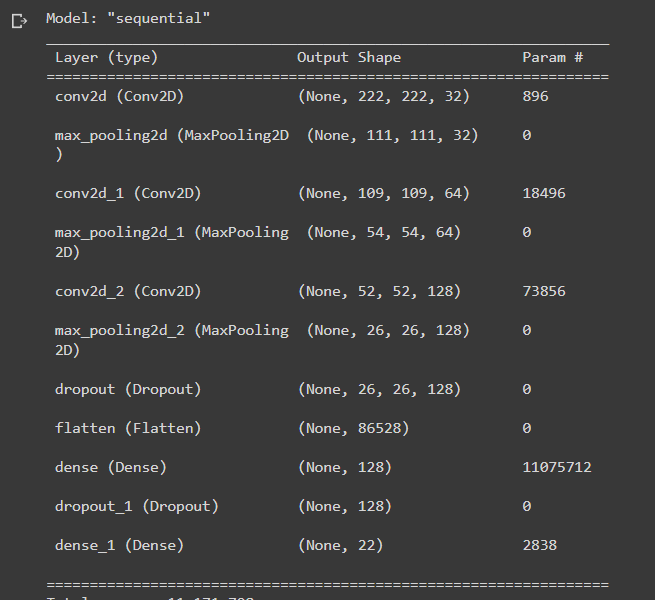
Pada blok 3, terdapat satu layer tambahan yaitu *dropout layer.* Layer ini berfungsi untuk mencegah terjadinya overfitting model. Overfitting terjadi ketika model sangat cocok dengan data latihnya. Berikut perhitungan pada blok 3, dengan jumlah kernel pada *convolution layer* sebanyak 128 buah.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

Karena diproses dengan MaxPooling dengan filter matriks (2 x 2), maka menghasilkan citra dengan dimensi *26 x 26 x 128*. Selanjutnya masuk tahap *flatten,* yaitu proses mengubah feature map hasil operasi pada blok 1, 2, dan 3 menjadi array satu dimensi. Perhitungannya adalah dengan sebagai berikut:

|  |  |
| --- | --- |
|  |  |
|  |  |

Berikutnya masuk fully connected layer dengan jumlah node sebanyak 128 buah. Lalu dilakukan dropout yang ke-2, sebelum masuk pada output layer dengan 22 node. Berikut adalah layer-layer CNN disajikan pada Gambar 4.2.



**Gambar 4.2 Layer CNN**

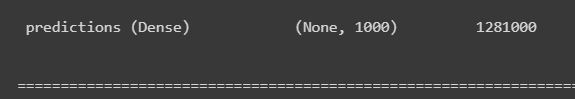
Proses terakhir dalam modeling percobaan 1 ini adalah training model tersebut dengan dataset yang sebelumnya sudah dibagi seperti uraian pada sub sub bab 4.2.1. Pada training model ini, dilakukan sebanyak 20 iterasi (epochs) dengan durasi waktu (ETA: 45 menit) menggunakan Google Colaboratory. Berikut adalah hasil dari training model CNN percobaan 1:

|  |  |
| --- | --- |
| *Loss* | : 0,1998 |
| *Accuracy* | : 0,9308 |
| *Validation loss* | : 2,4045 |
| *Validation accuracy* | : 0,4958 |

Model pada percobaan 1 mengalami *overfitting*. Dapat dilihat pada nilai akurasi pada data training (accuracy) sebesar 0,9308 tetapi akurasi pada data validation (validation accuracy) sebesar 2,4045. Dropout layer pada model CNN ini belum mampu mencegah terjadinya hal tersebut. Beberapa penyebab overfitting model antara lain: jumlah layer kurang banyak (tidak sebanding dengan jumlah data latih yang jumlahnya ribuan), atau variasi dataset yang tidak baik (cenderung sama). Model hasil percobaan 1 ini tetap bisa digunakan untuk mengidentifikasi data uji. Namun, mengingat akurasinya yang rendah dan terjadi overfitting, maka penulis melakukan percobaan kedua untuk mendapatkan model yang lebih baik dan ideal.

Custom Model CNN

Pada percobaan kedua ini, modeling dilakukan dengan memanfaatkan base model MobileNet-v2. Model tersebut merupakan model yang sudah ditraining berkali-kali, sehingga sudah “cerdas”. MobileNet-v2 ini mampu mengklasifikasikan 1000 jenis citra, karena pada output layer mempunyai 1.000 node seperti pada Gambar 4.x berikut. *Dataset* MobileNet-v2 adalah *imagenet*.



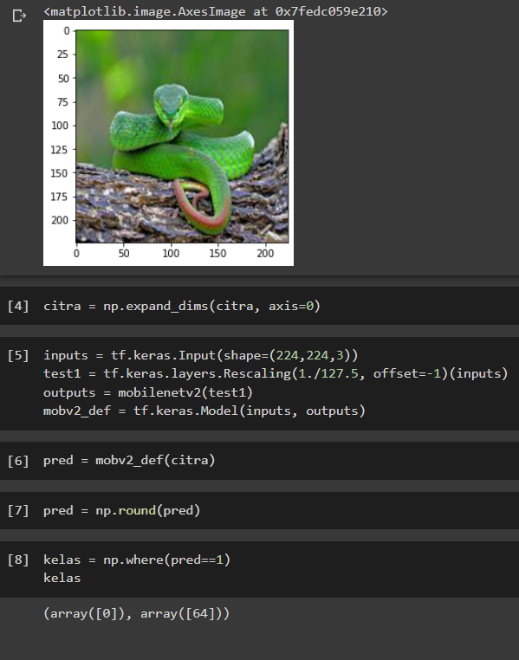
**Gambar 4.x Output Layer Default MobileNet-v2**

Penulis mencoba melakukan pengujian tanpa melakukan training pada base model MobileNet-v2 ini. Tujuannya adalah untuk mengetahui apakah model ini bisa mengidentifikasi citra uji (wayang kulit). Dan membandingkan dengan citra uji dengan gambar-gambar umum, seperti bunga, kendaraan, atau hewan. Berikut adalah hasilnya:



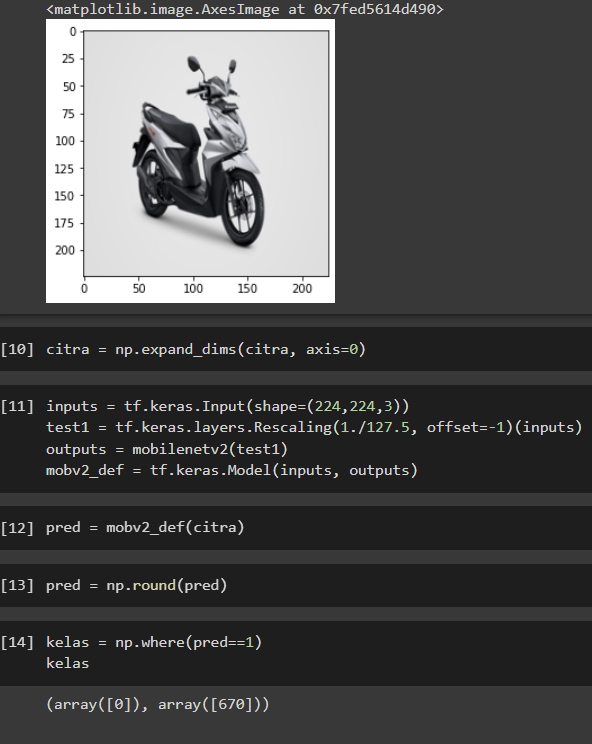
**Gambar 4.x Hasil Prediksi 1**

Pada Gambar 4.x citra uji berhasil diidentifikasi, bahwa citra uji tersebut mirip atau cocok dengan IMAGENET 1000 Class List pada array indeks ke-985. Berdasarkan dataset Imagenet, gambar pada indeks array ke-985 yaitu *daisy* (jenis bunga).



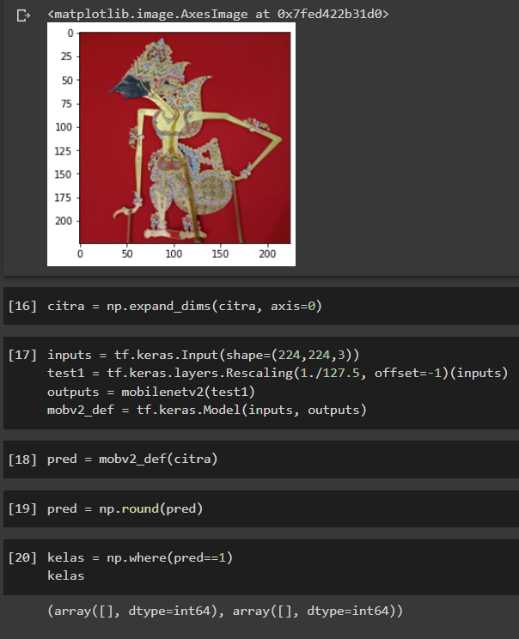
**Gambar 4.x Hasil Prediksi 2**

Pada Gambar 4.x citra uji berhasil diidentifikasi, bahwa citra uji tersebut mirip atau cocok dengan IMAGENET 1000 Class List pada array indeks ke-64. Berdasarkan dataset Imagenet, gambar pada indeks array ke-64 yaitu *green mamba* (jenis ular).



**Gambar 4.x Hasil Prediksi 3**

Pada Gambar 4.x citra uji berhasil diidentifikasi, bahwa citra uji tersebut mirip atau cocok dengan IMAGENET 1000 Class List pada array indeks ke-670. Berdasarkan dataset Imagenet, gambar pada indeks array ke-670 yaitu *motor scooter, scooter* (jenis kendaraan).



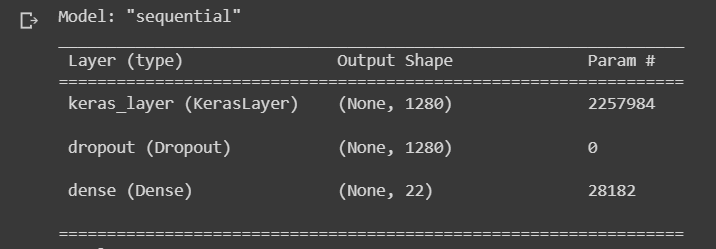
**Gambar 4.x Hasil Prediksi 4**

Sedangkan pada Gambar 4.x citra uji tidak berhasil diidentifikasi. IMAGENET 1000 Class List tidak terdapat gambar yang cocok dengan citra uji (wayang kulit). Oleh sebab itu, pada modeling percobaan kedua ini penulis akan melakukan modifikasi atau kombinasi antara base model MobileNet-v2 dengan layer-layer CNN classic (seperti percobaan pertama) sehingga tercipta custom model CNN. Pada modeling ini, penulis menambahkan satu layer sebelum base model MobileNet-v2 dan beberapa layer di akhir. Adapun layer-layer yang ditambahkan pada base model MobileNet-v2, antara lain: *rescaling, dropout layer,* dan *dense (output layer).*

Proses *rescale = 1./255* akan mengonversi piksel dalam rentang [0, 255] ke rentang [0, 1]. Proses ini juga disebut normalisasi *input*. Menskalakan setiap gambar ke rentang yang sama [0, 1] akan membuat gambar berkontribusi lebih merata terhadap total *loss*. *Neural network* memiliki peluang lebih tinggi untuk konvergen karena membuat koefisien dalam kisaran [0, 1] dibandingkan dengan [0, 255] sehingga membantu model memproses *input* lebih cepat.

Layer selanjutnya yang ditambahkan pada base model MobileNet-v2 adalah dropout. Seperti yang sudah dijelaskan pada percobaan pertama, fungsi ini mencegah terjadinya overfitting model (seperti hasil model CNN percobaan 1). Presentase dropout layer yang digunakan yaitu 0.4 atau 40%.

Pada output layer, dense disesuaikan dengan jumlah label atau kelas pada penelitian ini yang berjumlah 22 kelas dengan fungsi aktivasi *softmax*. Output shape juga diinisialisasi sebesar 1280 seperti pada Gambar 4.x berikut.



**Gambar 4.x Arsitektur Custom Model CNN**

Mekanisme kerja custom model CNN ini secara prinsip sama seperti arsitektur CNN pada percobaan 1. Terdapat convolution layer, pooling layer, flatten, dropout, dll. Input shape citra pada modeling percobaan 2 ini berdimensi *224 x 224 x 3 (RGB)*, dan hasil akhirnya menghasilkan dimensi *7 x 7 x 1280*.

Urutan terakhir dari modeling percobaan 2 ini yaitu training custom model CNN dengan dataset wayang kulit hasil *preprocessing*. Mula-mula dilakukan sebanyak 20 iterasi (epochs), dengan durasi (ETA: 25 menit) dan menghasilkan informasi sebagai berikut:

|  |  |
| --- | --- |
| *Loss* | : 0,6874 |
| *Accuracy* | : 0,7802 |
| *Validation loss* | : 0,6412 |
| *Validation accuracy* | : 0,8056 |

Hasil dari 20 kali iterasi training sudah cukup baik. Namun, dilakukan iterasi lagi sebanyak 100 kali (ETA: 30 menit), dengan tujuan menurunkan nilai *loss* dan *validation loss* yang masih cukup besar. Berikut adalah hasilnya:

|  |  |
| --- | --- |
| *Loss* | : 0,5386 |
| *Accuracy* | : 0,8216 |
| *Validation loss* | : 0,5828 |
| *Validation accuracy* | : 0,8140 |

Terdapat peningkatan *accuracy dan validation accuracy*, dan penurunan pada *loss dan validation loss*. Dengan demikian, custom model CNN ini yang akan diimplementasikan pada proses identifikasi citra wayang kulit.

### Tahap Identifikasi Data Uji

Pada tahap identifikasi data uji, langkah yang pertama adalah proses *load* custom model CNN hasil modeling percobaan 2, dengan ekstensi *.h5*. Selanjutnya, inisialisasi label bertipe *array* dengan nama-nama karakter wayang yang berjumlah 22, antara lain: *abimanyu, anoman, arjuna, bagong, baladewa, bima, buta, cakil, durna, dursasana, duryudana, gareng, gatotkaca, karna, kresna, nakula\_sadewa, patih\_sabrang, petruk, puntadewa, semar, sengkuni,* dan *togog.*

Ketika program dijalankan, maka system akan meminta *user* mengunggah file (citra wayang kulit). Sebelum masuk proses identifikasi, citra uji tersebut akan dideteksi tepi (seperti *preprocessing*). Proses tersebut meliputi mengubah ukuran citra menjadi 224 x 224, konversi ke citra *grayscale*, dilakukan deteksi tepi, citra diubah ke array dengan sintaks *img\_to\_array()* sehingga citra memiliki dimensi (height x width x channels), selanjutnya yaitu menambah dimensi citra menjadi (1 x height x width x channels). Hal itu dilakukan karena model akan membaca data uji dengan dimensi tersebut. Langkah terakhir yaitu *rescaling* (1./255) sehingga mempunyai rentang [0, 1].

Setelah citra uji selesai di-*preprocessing,* maka citra siap untuk diidentifikasi oleh custom model CNN. Model akan menampilkan label (nama karakter wayang kulit) dan presentasenya.

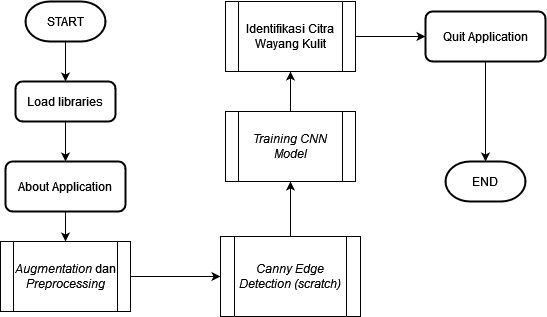
### Tahap Visualisasi Hasil

Pada tahap ini, hasil identifikasi citra uji akan ditampilkan pada sebuah *page* atau *form*. Label atau teks dengan warna biru menandakan hasil identifikasi *benar*. Sedangkan label atau teks yang berwarna merah, menunjukkan hasil prediksi *salah*. Selain itu, citra uji juga akan ditampilkan, agar lebih memudahkan *user* dalam mengenali nama-nama karakter wayang kulit ini. Detail informasi terkait visualisasi hasil dapat dilihat pada rancangan layar Gambar 3.x dan tampilan layar visualisasi Gambar 4.x.

## Flowchart Tahapan Metode

Flowchart merupakan suatu bagan atau simbol yang menggambarkan alur kerja atau urutan proses pada suatu program. Berikut adalah penjabaran flowchart dalam penelitian ini.

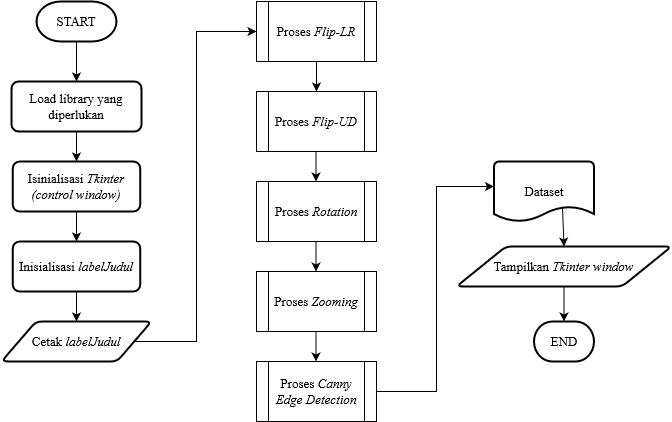
### Flowchart Keseluruhan Sistem



**Gambar 4.x Flowchart Keseluruhan Sistem Aplikasi**

Flowchart pada Gambar 4.x ini menjelaskan tahapan mengenai berjalannya sistem, mulai dari awal sampai dengan akhir, yaitu mendapatkan hasil identifikasi citra wayang kulit. Beberapa proses pada alur keseluruhan system ini yaitu proses *load library*, menu *about application* yang berisi detail informasi system aplikasi ini. Selanjutnya masuk tahap augmentasi dan *preprocessing* untuk menambah variasi dataset dan mengubah citra mentah menjadi citra hasil deteksi tepi. Canny Edge Detection akan khusus dijabarkan detail prosesnya. Setelah terbentuk *fix dataset* maka dilakukan proses training model CNN, dengan hasil akhir berupa custom model CNN berekstensi .*h5*. Berikutnya adalah tahap mengidentifikasi data uji pada custom model CNN yang sudah jadi. Hasil identifikasi akan ditampilkan berupa nama karakter wayang kulit beserta tingkat kemiripannya.

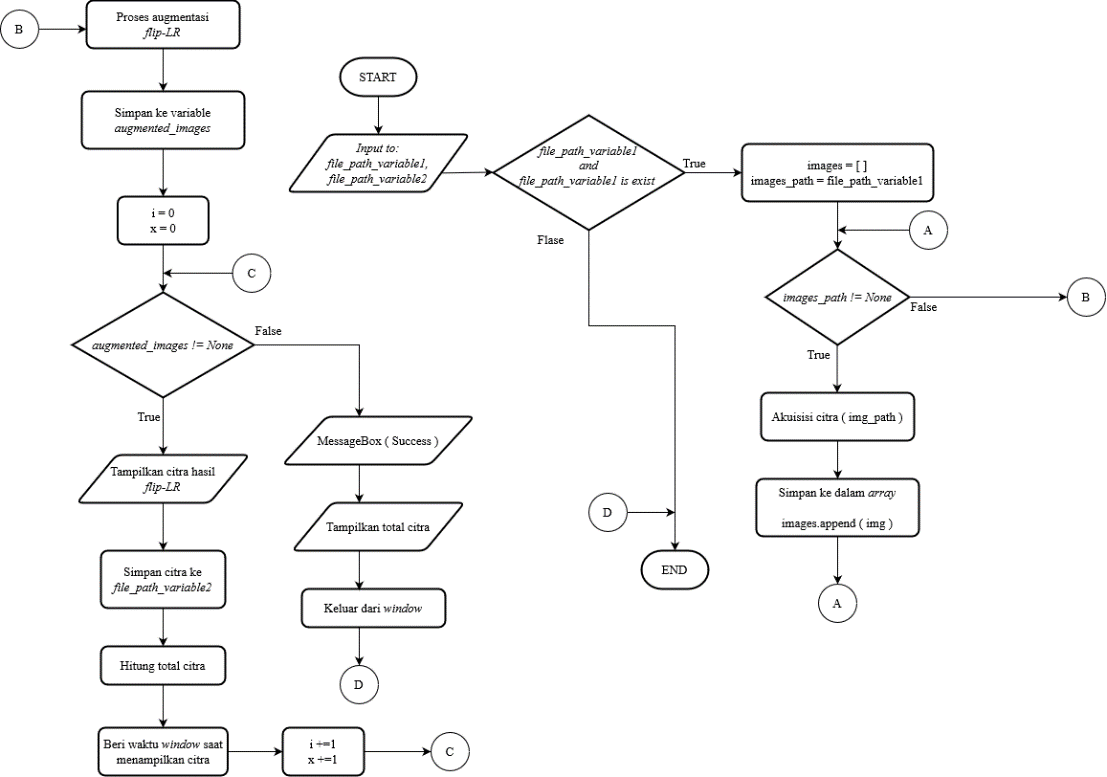
### Flowchart *Preprocessing*



**Gambar 4.x Flowchart Alur Preprocessing**

*Flowchart* pada Gambar 4.x menggambarkan alur proses dalam augmentasi dan *prepocessing* data. Tahap augmentasi meliputi *flip horizontal (LR), flip vertical (UD), rotation, zooming*. Sedangkan tahap *preprocessing* data merupakan proses deteksi tepi pada citra wayang kulit. Hasil dari kedua proses tersebut akan masuk ke dalam dataset untuk keperluan *training* model CNN.

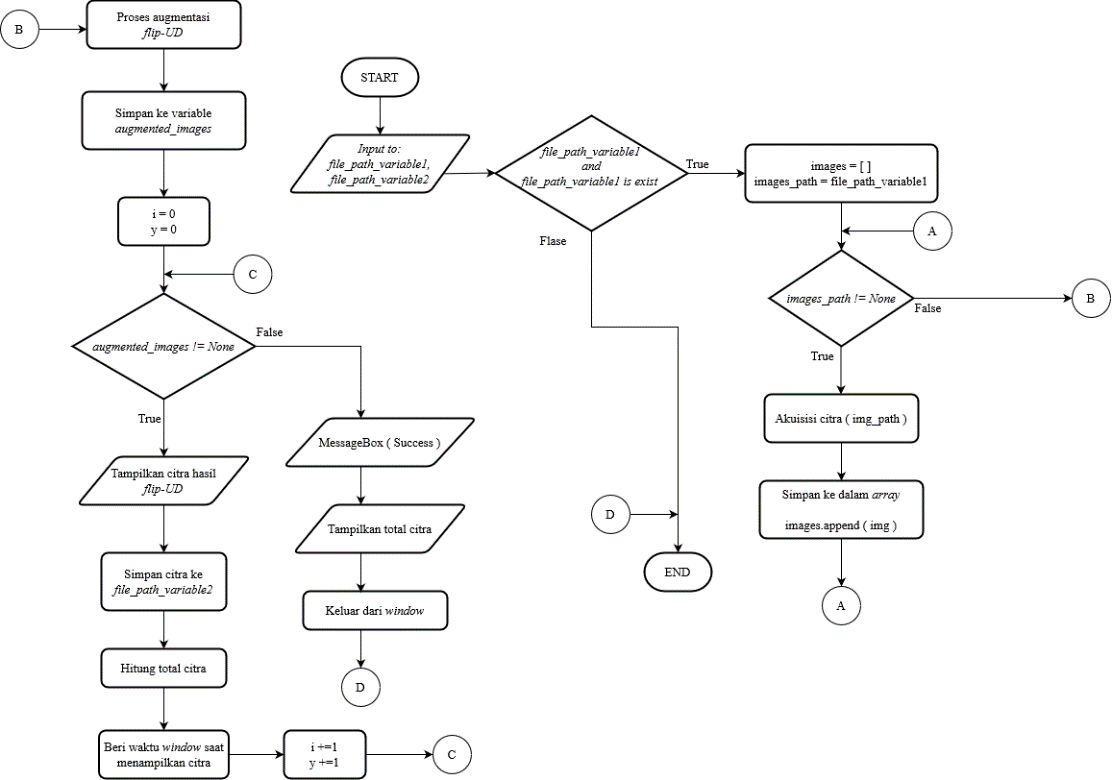
*Flowchart Flip Horizontal* (LR)



**Gambar 4.x Flowchart Proses Flip Horizontal (LR)**

Flowchart pada Gambar 4.x menggambarkan alur proses augmentasi data yaitu flip horizontal (LR). Diawali dengan memilih folder yang di dalamnya berisi kumpulan file citra wayang kulit yang akan diaugmentasi. Hasil dari citra flip horizontal akan disimpan pada folder yang sama (folder awal) dengan nama file yang berbeda, untuk kemudian diproses di tahap berikutnya.

Flowchart Flip Vertikal (UD)

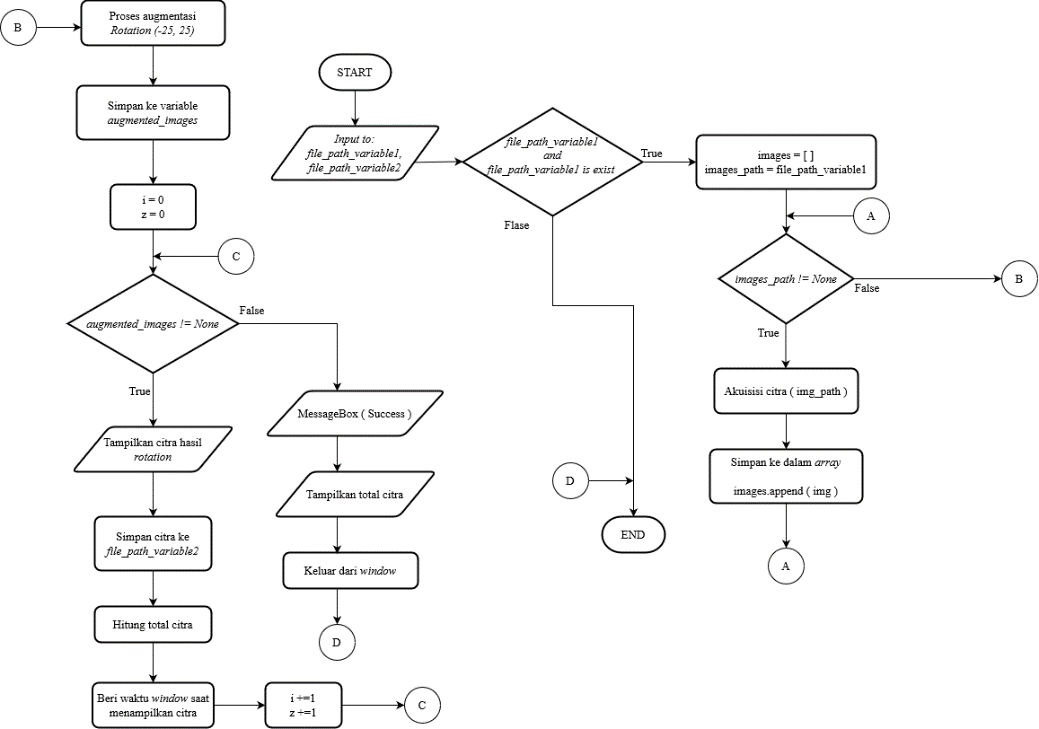


**Gambar 4.x Flowchart Proses Flip Vertical (UD)**

Flowchart pada Gambar 4.x menggambarkan alur proses augmentasi data yaitu flip vertical (UD). Diawali dengan memilih folder yang di dalamnya berisi kumpulan file citra wayang kulit yang akan diaugmentasi. Hasil dari citra flip vertical akan disimpan pada folder yang sama (folder awal) dengan nama file yang berbeda, untuk kemudian diproses di tahap berikutnya.

*Flowchart Rotation*

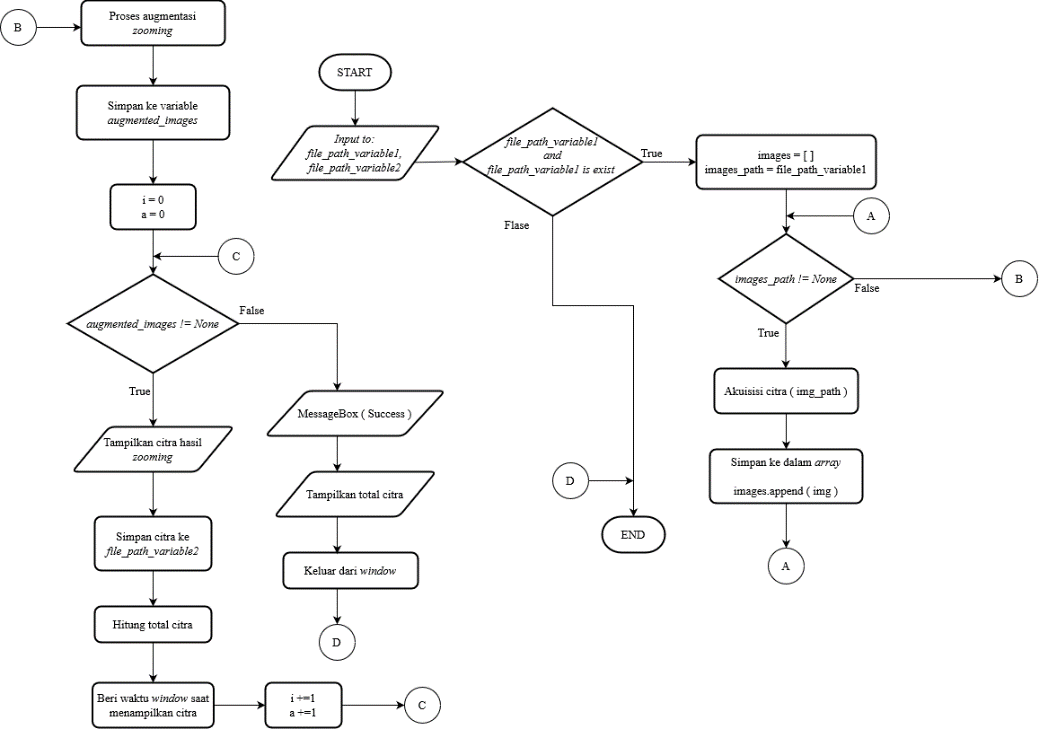
Flowchart pada Gambar 4.x menggambarkan alur proses augmentasi data yaitu rotation dengan sudut -25 sampai 25 derajat. Diawali dengan memilih folder yang di dalamnya berisi kumpulan file citra wayang kulit yang akan diaugmentasi. Hasil dari citra rotasi akan disimpan pada folder yang sama (folder awal) dengan nama file yang berbeda, untuk kemudian diproses di tahap berikutnya.



**Gambar 4.x Flowchart Proses Flip Vertical (UD)**

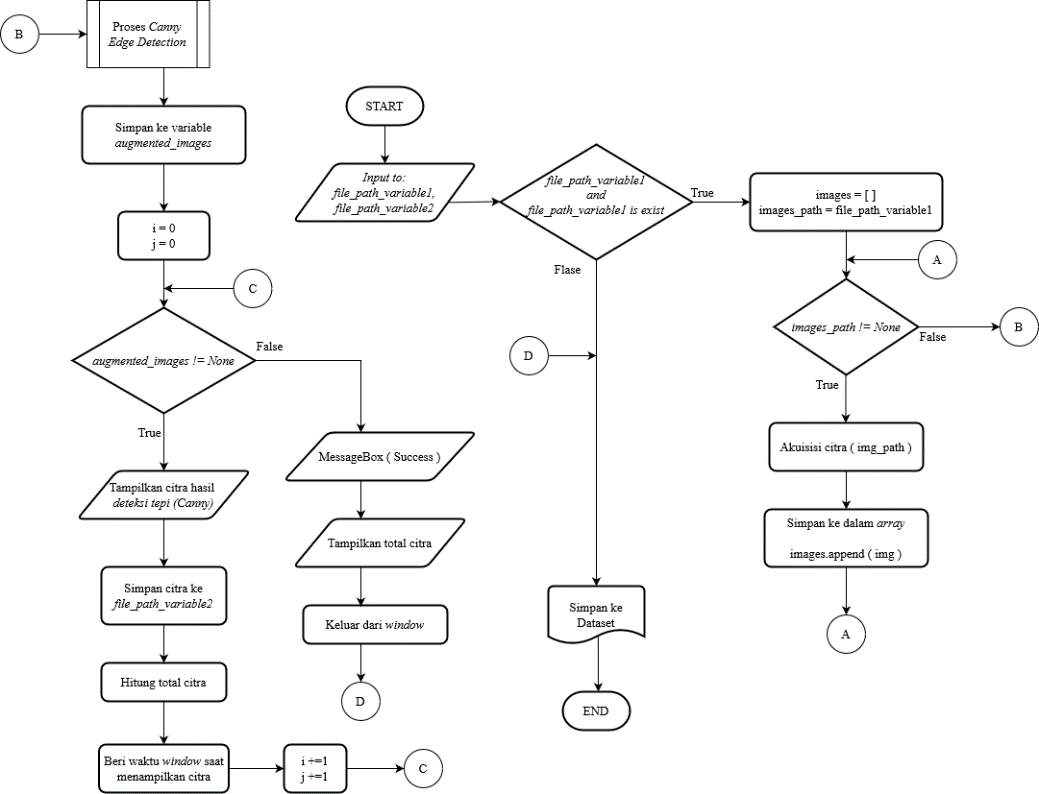
*Flowchart Zooming*

Flowchart pada Gambar 4.x menggambarkan alur proses augmentasi data yaitu zooming dengan presentase sebesar 40%, 100%, 80%, dan 120% pada setiap sumbunya. Diawali dengan memilih folder yang di dalamnya berisi kumpulan file citra wayang kulit yang akan diaugmentasi. Hasil dari citra zooming akan disimpan pada folder yang sama (folder awal) dengan nama file yang berbeda, untuk kemudian diproses di tahap berikutnya.



**Gambar 4.x Flowchart Proses Flip Vertical (UD)**

Flowchart Canny Edge Detection

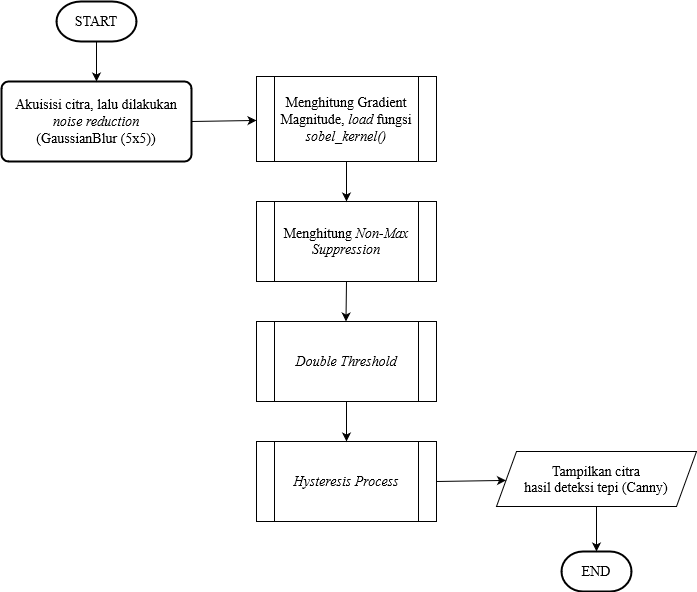


**Gambar 4.x Flowchart Proses Flip Vertical (UD)**

*Flowchart* pada Gambar 4.x menggambarkan alur proses *preprocessing* dengan *Canny Edge Detection*. Diawali dengan memilih folder yang di dalamnya berisi kumpulan file citra wayang kulit yang akan diaugmentasi. Hasil dari citra *edge detection* ini akan disimpan pada folder atau dataset, kemudian diunggah ke *Google Drive* untuk keperluan training model CNN. Pada penelitian ini, metode *Canny Edge Detection* akan dijabarkan lebih dalam seperti berikut.

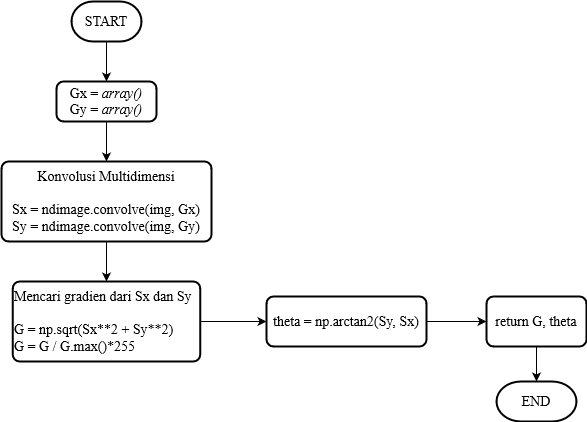
### Flowchart Canny Edge Detection (Scratch)

Pada Gambar 4.x merupakan alur proses atau flowchart metode Canny Edge Detection. Langkah pertama adalah mengakuisisi citra dan dihaluskan dengan GaussianBlurr (5 x5). Kemudian adalah menghitung *Gradient Magnitude, Non-Maximum Suppression,* menentukan *Double Threshold,* dan *Hysteresis Process.* Langkah-langkah tersebut dapat dilihat pada ilustrasi *flowchart* berikut.



**Gambar 4.x Flowchart Proses Flip Vertical (UD)**

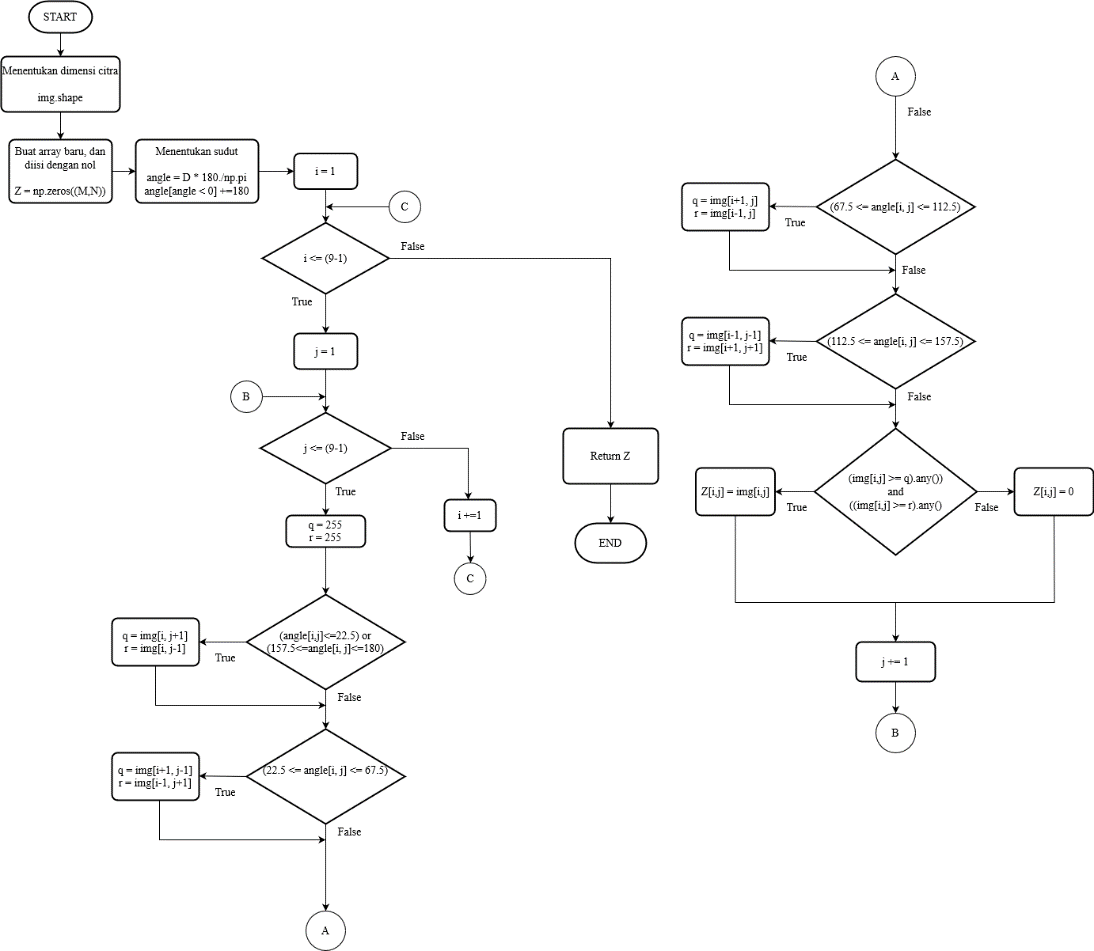
*Flowchart Calculate Gradient Magnitude*

**

**Gambar 4.x Flowchart Proses Flip Vertical (UD)**

Flowchart pada Gambar 4.x menggambarkan alur mencari *gradient magnitude* dan sudut (arah). Diawali inisialisasi Gx dan Gy yang berisi array angka-angka tertentu, lalu dihitung dengan operasi konvolusi sehingga menghasilkan Sx dan Sy. Selanjutnya, hasil dari operasi konvolusi tersebut menjadi variable saat mencari nilai gradient magnitude dan theta (sudut). Lalu nilai dikembalikan dengan perintah *return*.

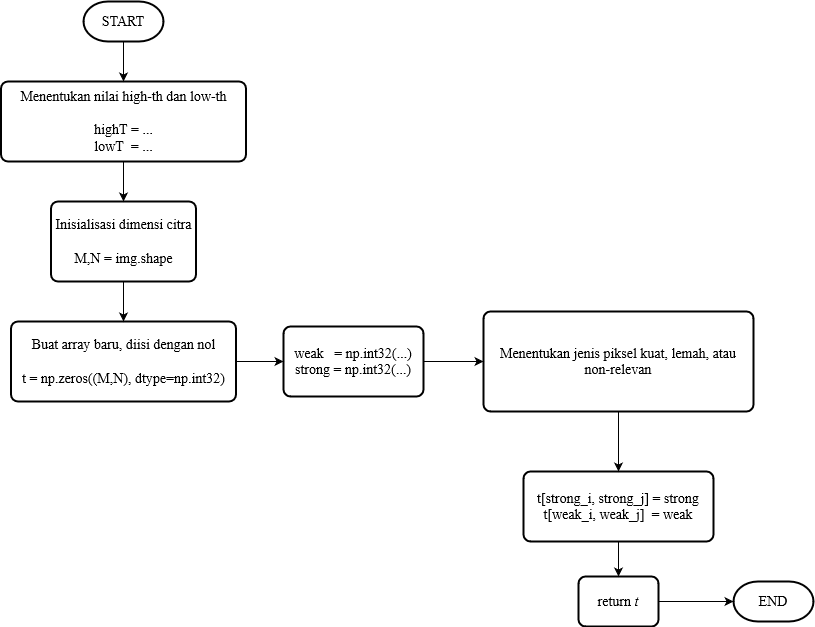
*Flowchart Non-Maximum Suppression*



**Gambar 4.x Flowchart Proses Flip Vertical (UD)**

Flowchart pada Gambar 4.x merupakan gambaran alur proses perhitungan non-maximum suppression. Konsep dari non-maximum suppression adalah mengecek piksel-piksel tetangga, apakah terdapat piksel kuat atau tidak. Jika terdapat piksel kuat, maka piksel yang dioperasi dianggap sebagai *edge*, dan jika tidak maka akan diberi nilai 0 (bukan *edge*).

*Flowchart Double Threshold*

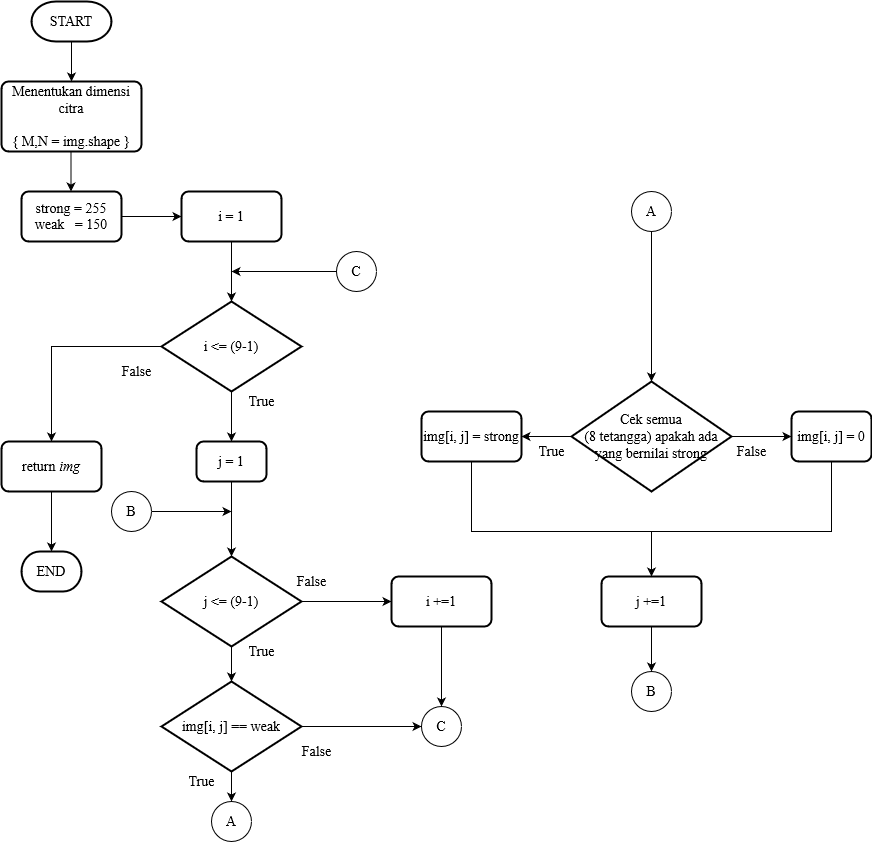


**Gambar 4.x Flowchart Proses Flip Vertical (UD)**

Flowchart pada Gambar 4.x merupakan gambaran proses double threshold. Pada proses ini akan ditentukan tiga jenis piksel, antara lain: piksel kuat, piksel lemah, dan piksel non-relevan. Piksel dengan kategori lemah selanjutnya akan dipastikan termasuk edge atau bukan pada operasi hysteresis.

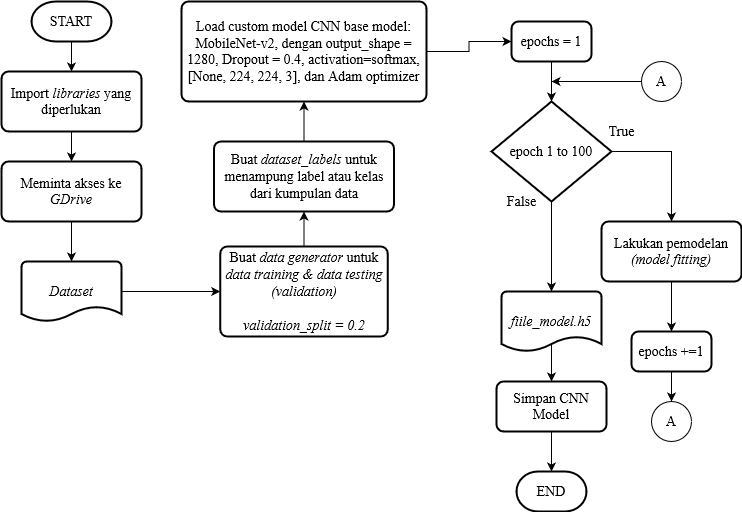
*Flowchart Hysteresis Process*

Flowchart pada Gambar 4.x merupakan gambaran alur proses hysteresis. Pada operasi hysteresis ini bertujuan untuk memastikan piksel-piksel dengan kategori lemah apakah bisa dianggap sebagai *edge* atau tidak. Langkahnya adalah dengan mengecek piksel-piksel tetangganya. Jika ada piksel kuat, maka dianggap sebagai *edge*. Dan jika tidak ada, maka bukan *edge*. Inilah akhir dari metode *Canny Edge Detection.*



**Gambar 4.x Flowchart Proses Flip Vertical (UD)**

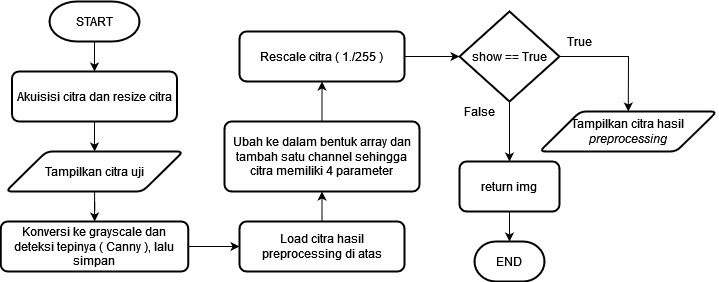
### Flowchart Training Model CNN



**Gambar 4.x Flowchart Proses Flip Vertical (UD)**

Flowchart pada Gambar 4.x menunjukkan alur proses ketika training custom model CNN (base model: MobileNet-v2). Langkah pertama adalah load libraries yang diperlukan, lalu mengakses dataset dari Google Drive. Berikutnya adalah splitting data dengan rasio 2:8 seperti pada penjelasan sub sub bab 3.2.3. Selanjutnya adalah inisialisasi label atau nama kelas (22 nama karakter wayang kulit). Langkah berikutnya yaitu modeling seperti pada penjelasan sub sub bab 4.2.4, kemudian training dengan 100 kali iterasi (epochs) sampai custom model CNN terbentuk (ekstensi file .*h5*).

### *Flowchart* Proses *Load Image*

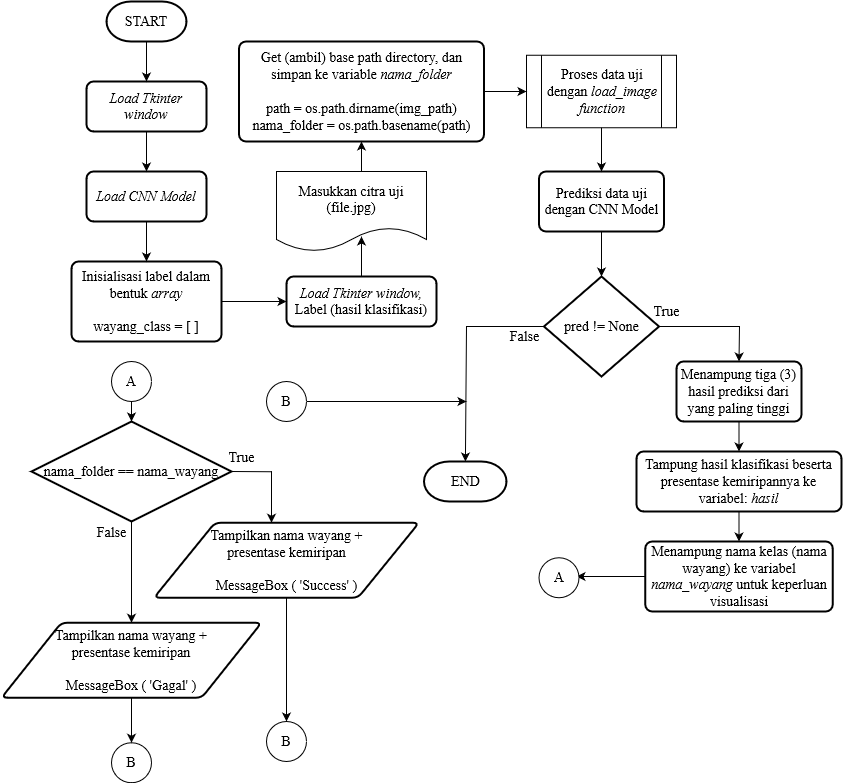


**Gambar 4.x Flowchart Proses Flip Vertical (UD)**

Flowchart pada Gambar 4.x merupakan gambaran proses load image atau preprocessing citra uji sebelum dilakukan proses identifikasi. Langkah-langkahnya antara lain: mengakuisisi dan mengubah dimensi citra uji, menampilkan hasilnya, mengonversi ke citra *grayscale* dan mendeteksi tepinya, mengubahnya ke citra dengan 4 parameter = [batch, height, width, channels], lalu rescaling ( 1./255 ), dan mengembalikan hasil citra pemrosesan tersebut untuk selanjutnya dilakukan proses identifikasi dengan custom model CNN.

### *Flowchart* Identifikasi Data Uji

Flowchart pada Gambar 4.x merupakan gambaran proses identifikasi data uji (citra wayang kulit *raw*). Citra uji tersebut diambil dari data uji. Pada proses identifikasi ini, custom model CNN akan mengklasifikasikan dan menampilkan tiga hasil prediksi teratas. Namun, nama karakter wayang yang ditampilkan hanya satu (yang paling atas).



**Gambar 4.x Flowchart Proses Flip Vertical (UD)**

## Algoritma Tahapan Metode

### Algo1

### Algo 2

## Pengujian

## Tampilan Layar Aplikasi

### Tampilan Layar Menu Utama

### Dst

# PENUTUP

## Kesimpulan

## Saran

# LAMPIRAN

**DAFTAR PUSTAKA**

Anwar, G. A., & Riminarsih, D. (2019). Klasifikasi Citra Genus Panthera Menggunakan Metiode Convolutional Neural Network (CNN). *Jurnal Ilmiah Informatika Komputer*, *24*(3), 220–228. https://doi.org/10.35760/ik.2019.v24i3.2364

Bowo, T. A., Syaputra, H., & Akbar, M. (2020). Penerapan Algoritma Convolutional Neural Network Untuk Klasifikasi Motif Citra Batik Solo. In *Journal of Software Engineering Ampera* (Vol. 1, Issue 2). https://journal-computing.org/index.php/journal-sea/index

Dadhich, A. (2018). *Practical Computer Vision Extract insightful information from images using TensorFlow, Keras, and OpenCV*. Packt Publishing.

Emerson, K. A. (2017). *PEMBAHARUAN WAYANG UNTUK PENONTON TERKINI Gaya Pakeliran Garap Semalam Sajian Dramatik Ki Purbo Asmoro, 1989 - 2017* (R. Wiratama, Ed.). ISI PRESS.

Farid Naufal, M., & Ferdiana Kusuma, S. (2021). PENDETEKSI CITRA MASKER WAJAH MENGGUNAKAN CNN DAN TRANSFER LEARNING. *Jurnal Teknologi Informasi Dan Ilmu Komputer (JTIIK)*, *8*(6), 1293–1300. https://doi.org/10.25126/jtiik.202185201

Fitra Maulana, F., & Rochmawati, N. (2019). Klasifikasi Citra Buah Menggunakan Convolutional Neural Network. *Journal of Informatics and Computer Science*, *1*(2), 104–108.

Gultom, Y., Arymurthy, A. M., & Masikome, R. J. (2018). Batik Classification using Deep Convolutional Network Transfer Learning. *Jurnal Ilmu Komputer Dan Informasi*, *11*(2), 59. https://doi.org/10.21609/jiki.v11i2.507

Indraswari, R., Herulambang, W., & Rokhana, R. (2022). *Deteksi Penyakit Mata Pada Citra Fundus Menggunakan Convolutional Neural Network (CNN)* (Vol. 21, Issue 2). https://www.kaggle.com/datasets/jr2ngb/cataractdataset

Mehindra Prasmatio, R., Rahmat, B., & Yuniar, I. (2020). Deteksi dan Pengenalan Ikan Menggunakan Algoritma Convolutional Neural Network. In *Jurnal Informatika dan Sistem Informasi (JIFoSI)* (Vol. 1, Issue 2).

Mufti, A. (2015). RANCANGAN LAYAR SEBAGAI ALAT BANTU PENDEWASA INTERAKSI MANUSIA DENGAN KOMPUTER. In *Faktor Exacta* (Vol. 8, Issue 2).

Prasetyawan, D., & Informatika UIN Sunan Kalijaga Yogyakarta Jl Marsda Adi Sucipto Yogyakarta, M. (2020). Penentuan Emosi pada Video dengan Convolutional Neural Network. In *JISKa* (Vol. 5, Issue 1). MEI.

Primartha, R. (2018). *BELAJAR MACHINE LEARNING TEORI DAN PRAKTIK*. Informatika Bandung.

Primartha, R. (2021). *Algoritma Machine Learning* (R. S. Wahono, Ed.). Informatika.

Raschka, S., & Mirjalili, V. (2017). *Pyhton Machine Learning Second Edition* (2nd ed.). Packt Publishing Ltd.

Rasywir, E., Sinaga, R., Pratama, Y., Dinamika, U., & Jambi, B. (2020). Analisis dan Implementasi Diagnosis Penyakit Sawit dengan Metode Convolutional Neural Network (CNN). *Paradigma – Jurnal Informatika Dan Komputer*, *22*(2). https://doi.org/10.31294/p.v21i2

Resa Arif Yudianto, M., Kusrini, & al Fatta, H. (2020). *Analisis Pengaruh Tingkat Akurasi Klasifikasi Citra Wayang Dengan Algoritma Convolutional Neural Network*.

Sandy, B., Siahaan, J. K., Permana, P., & Muhathir, \*. (2019). Klasifikasi Citra Wayang Dengan Menggunakan Metode k-NN & GLCM. In *Prosiding Seminar Nasional Teknologi Informatika* (Vol. 2).

Sehairi, K., Chouireb, F., & Meunier, J. (2017). Comparative study of motion detection methods for video surveillance systems. *Journal of Electronic Imaging*, *26*(2), 023025. https://doi.org/10.1117/1.jei.26.2.023025

Sukatmi. (2017). *Perbandingan Deteksi Tepi Citra Digital dengan Metode Prewitt, Sobel dan Canny*.

Susanto, A., Sari, C. A., Rosal, D., Setiadi, I. M., & Rachmawanto, E. H. (2017). PERLINDUNGAN HAK CIPTA PADA CITRA DIGITAL MENGGUNAKAN LEAST SIGNIFICANT BIT BERBASIS DETEKSI TEPI CANNY. *Jurnal SIMETRIS*, *8*.

Susanto, A., Utomo, I., & Mulyono, W. (2019). *Rekognisi Wayang Kulit Menggunakan Jaringan Syaraf Tiruan*.

Teknik Elektro, J., & Wega Intyanto, G. (2021). *Klasifikasi Citra Bunga dengan Menggunakan Deep Learning: CNN (Convolution Neural Network)*.

Wikarta, A., Effendi, M. K., & Pramono, A. S. (2021). Sistem Pendeteksi Masker pada Pengemudi Kendaraan Menggunakan Kecerdasan Artifisial. *JEPIN (Jurnal Edukasi Dan Penelitian Informatika)* , *7*(2), 250–254.

Wulandari, I., Yasin, H., & Widiharih, T. (2020). Klasifikasi Citra Digital Bumbu dan Rempah Dengan ALgoritma Convolutional Neural Network (CNN). *JURNAL GAUSSIAN*, *9*(3), 273–282. https://ejournal3.undip.ac.id/index.php/gaussian/

[https://www.indonesia.travel/id/id/ide-liburan/diakui-sebagai-warisan-unesco-ini-5-hal-menarik-seputar-wayang-kulit (8](https://www.indonesia.travel/id/id/ide-liburan/diakui-sebagai-warisan-unesco-ini-5-hal-menarik-seputar-wayang-kulit%20(8) April 2022 | 14.15)

<https://opencv.org/about/> (8 april 2022 | 22.20)

[https://www.ibm.com/cloud/learn/supervised-learning (11](https://www.ibm.com/cloud/learn/supervised-learning%20(11) april 2022 | 13.10)

[https://www.mathworks.com/help/deeplearning/ref/mobilenetv2.html (1](https://www.mathworks.com/help/deeplearning/ref/mobilenetv2.html%20(1) juni 2022 | 22:10)

<https://medium.com/nodeflux/mobilenet-deteksi-objek-pada-platform-mobile-bbbf3806e4b3> (tentang arsitektur mbnetv2) 12 juni 2022 00.30

<https://oden.io/glossary/model-training/> (tentang training model) 12 juni 2022 20:00

<https://techdayhq.com/community/articles/top-5-machine-learning-libraries-today> (library - “ -)